MICRO

THE 6502 JOURNAL

No. 25

JUNE 1980

$2.00

TRENDCOM 200

High-Speed Intelligent Printe

```
100 INPUT "TIMES PAGE 1 OR 2",X
110 IF X=1 THEN POKE 2,32
120 IF X=2 THEN POKE 2,04
150 INPUT "SHOULD LIGHT ON SCREEN EQUAL WHITE ON PRINTER (Y/N)?",Y$
160 IF Y$="Y" THEN POKE 1145,0
170 IF Y$="N" THEN POKE 1145,1
200 INPUT "PRINT UNIDIRECTIONAL (Y/N)",U$
210 IF U$="Y" THEN POKE 1144,0
220 IF U$="N" THEN POKE 1144,1
230 IF C#40 THEN 250
240 X$="N": GOTO 260
250 INPUT "BLOW UP TIMES 2 (Y/N)",X$
260 IF X$="N" THEN 290
270 POKE 1147,1:R$="Y": GOTO 310
290 POKE 1147,0
295 IF C#40 THEN 300
296 R$="Y": GOTO 310
300 INPUT "ROTATE 90 DEGREES (Y/N)",R$
```

40 characters-per-second

80 characters per line

Upper and lower case

Continuous graphics at 60 dots per inch

Microprocessor controlled

Bidirectional look-ahead printing

Automatic "wrap-around"

Quiet operation

The Trendcom 200 is a high speed thermal printer offering the combination of text printing at 80 characters per line and continuous graphics at 60 dots per inch. In the text mode, upper and lower case data are printed at 40 characters per second. The 5 x 7 characters provide clear readable copy on white paper; no hard to find, hard to read aluminized paper.

In the graphics mode, seven bits of each byte correspond to the seven dots in each of the 480 print positions per line. Since the computer driving the printer has full control over every print position, it can print graphs, bar charts, line drawings, even special and foreign language symbols. Despite its low cost, the Trendcom 200 is a true intelligent printer with full line buffering and bidirectional look-ahead printing. After one line has been printed left to right, the internal microprocessor examines the next line to choose the shortest print direction. The microprocessor also provides a built-in self-test mode for easy verification of proper operation.

High reliability is designed in: The thick film thermal print head has a life expectancy of 100,000,000 characters. Two DC stepping motors provide positive control of the print head and the paper drive, the printer's only driven parts. The absence of gears and solenoids also makes the printer extremely quiet; the only noise is the rustling of the paper advancing.

——— **Skyles PAL-80 printer(s) complete** with 2½ foot interface cable to attach to my PET at **$675.00 each.*** (Plus $10.00 shipping and handling.) I also will receive a test and graphics demonstration tape at no additional charge and over 150 feet of 8½ inch wide black on white thermal paper.

——— rolls of **8½ inch wide** by 85 ft. long thermal paper (black ink) at $5.00 each; or cartons at 10@ $45.00.

Visa, Mastercharge orders call (800) 538-3083

California orders please call (408) 257-9140

Skyles Electric Works
231E South Whisman Road MountainView,CA 94041 (415) 965-1735

SYBEX BRIDGES THE GAP

$12.95
ref. C202

A comprehensive introduction to microprocessor programming, using the 6502. From basic concepts to advanced data structures. Complete explanation of internal register and bus operation. The basic 6502 textbook.

$12.95
ref. D302

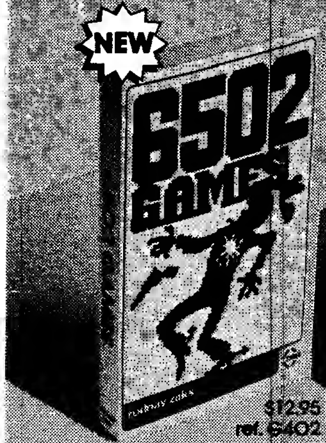
How to connect with the outside world and implement practical applications. Programs and circuits are presented, from home control to industrial applications, including analog-digital conversion. The I/O book for the 6802.

$10.95
ref. C201

From the ground up: the components (ROM, RAM, MPU, UART, PIO), their interconnect, applications, programming, system development. Used by educational institutions worldwide. "The basic text on microprocessors."

$14.95
ref. C207

How to connect a microprocessor to the outside world. Includes the peripherals, from keyboard to CRT and floppy disk, as well as A/D conversion, standard buses (S100 to IEEE 488) and basic troubleshooting.



$12.95
ref. 6402

An educational text on advanced programming techniques. Includes programs and hardware design for a variety of games, ranging from music to mastermind or tic-tac-toe. "Complex algorithms can be fun."

$14.95
ref. C281

A comprehensive introducion to programming the Z80 family. From basic programming to interrupts and time-sharing, including input-output techniques and common utility programs. A self-contained text.

$14.95
ref. C280

A comprehensive introduction to microprocessor programming, using the Z80. From basic concepts to advanced data structures. Complete explanation of internal register and bus operation. "The basic Z80 textbook."

SYBEX

2344 Sixth Street
Berkeley, Ca. 94710

TO ORDER:
By Phone: (415) 848-8233. Visa, MC, American Express.
By mail: Indicate quantity desired. Prepayment required.
Shipping: Add $1.50 per Book (UPS) or 75¢ (4th Class-allow 4 weeks delivery).
Tax: in California please add tax.

Please send me the following:
____
____
Name____________________
Company__________________
Address__________________
City__________ State____ Zip______
Amount enclosed______________
Charge my: □ Visa □ Master Charge
□ American Express
Card number________________
Expiration date______________
Signature______________ Date______
□ Send me your detailed catalog.

AVAILABLE AT BOOK AND COMPUTER STORES EVERYWHERE

MICRO™

June 1980
Issue Number 25

Staff

Editor/Publisher
Robert M. Tripp

Associate Editor
Mary Ann Curtis

Circulation Manager
Carol A. Stark

Art/Advertising Coordinator
Terry Spillane

Accounting
Diane Williamson

Comptroller
Donna M. Tripp

Production Assistant
L. Catherine Bland

MICRO™ is published monthly by:
MICRO INK, Inc., Chelmsford, MA 01824
Second Class postage paid at:
Chelmsford, MA 01824
Publication Number: COTR 395770

Subscription rates: U.S.	$15.00 per year
Foreign surface mail	$18.00 per year
Central America air	$27.00 per year
Europe/Japan air	$33.00 per year
Other air mail	$39.00 per year

For back issues, subscriptions, change of address or other information, write to:

MICRO, P.O. Box 6502,
Chelmsford, MA 01824
or call
617/256-5515

5 **Editorial**
MICRO through the ages
by Robert M. Tripp

7 **A Little Plus For Your Apple II**
Add some Apple II Plus features to your Apple II
by Craig Peterson

11 **Lower Case Lister**
Solution to a listing problems on PET printers
by Rev. James Strasma

15 **Put Your Hooks Into OSI BASIC**
How to add your own code to ROM Basic
by Edward H. Carlson

19 **SYM-1 BASIC Pack Program**
A program to remove REMs from your Basic
by George H. Wells, Jr.

23 **Share Your AIM Programs**
Merge comments with the AIM dissassembler
by Jody Nelis

35 **Introducing the Atari 800**
An overview of this new microcomputer
by William L. Colsher

37 **APPLE II Integer BASIC Program List by Page**
A list program to display pages of Basic
by Dave Partyka

42 **OSI's Small Systems Journal**
A series of notes on OSI based systems
by the OSI Staff

47 **BASIC and Machine Language Transfers with Micromodem II**
How to use the modem with the Apple II
by George J. Dombrowski

49 **PET-16**
The Apple II Sweet-16 modified for the PET
by Rev. James Strasma

53 **Slide Show for the SYM**
Use the Apple II Slide Show on the SYM
by David P. Kemp

57 **Hypocycloids on the 540**
An update for the OSI 540 boards
by E.D. Morris

59 **TRACER: A Debugging Tool for the APPLE II**
Some improvements to the Step/Trace operations
by R. Kovacs

64 **'Stop That PET' - Update**
Program to 'Stop That PET' for the new PET ROMs
by George R. Gaukel

65 **6502 Resource Update**
An updated listing of publications for the 6502
by Dr. William R. Dial

68 **MICRO Club Circuit**
Information about 6502 oriented clubs
by Mike Rowe

71 **The MICRO Software Catalog: XXI**
Continuing software product announcements
by Mike Rowe

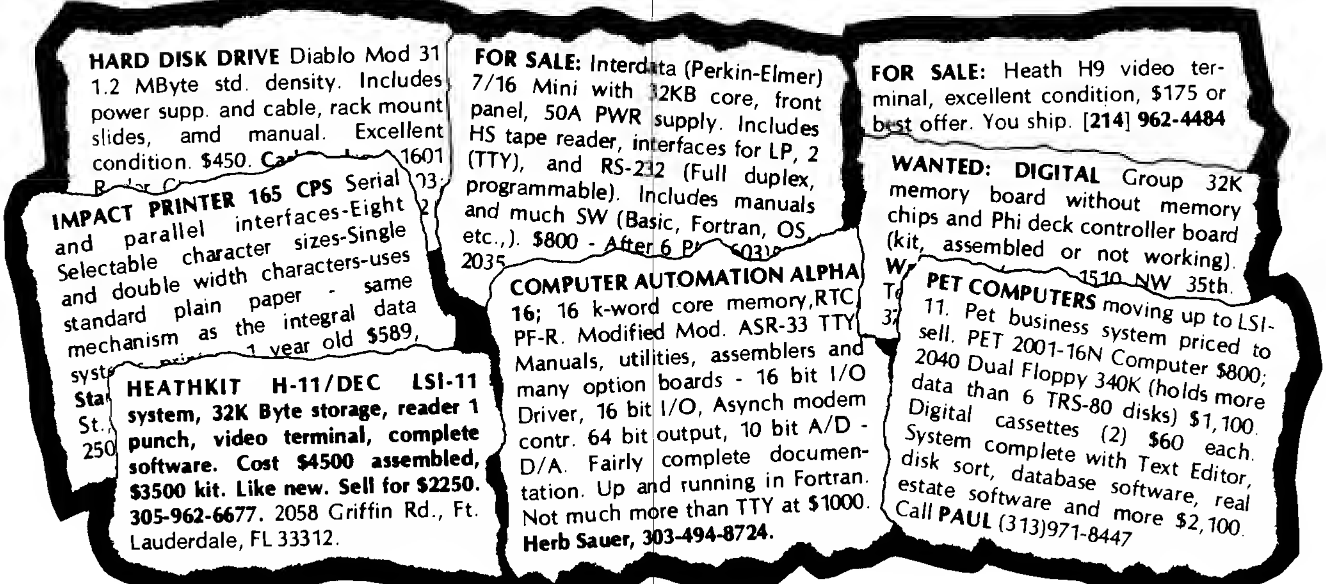
75 **6502 Bibliography: Part XXI**
Continuing coverage of 6502 related periodicals
by Dr. William R. Dial

79 **Advertisers' Index**

Attention

BARGAIN HUNTERS

Receive Hundreds of Classified Ads Like These Every Month



HARD DISK DRIVE Diablo Mod 31 1.2 MByte std. density. Includes power supp. and cable, rack mount slides, amd manual. Excellent condition. $450. Ca 1601 R 03; 2

IMPACT PRINTER 165 CPS Serial and parallel interfaces-Eight Selectable character sizes-Single and double width characters-uses standard plain paper - same mechanism as the integral data 1 year old $589, syst Sta St., 250

HEATHKIT H-11/DEC LSI-11 system, 32K Byte storage, reader 1 punch, video terminal, complete software. Cost $4500 assembled, $3500 kit. Like new. Sell for $2250. 305-962-6677. 2058 Griffin Rd., Ft. Lauderdale, FL 33312.

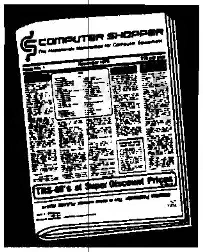
FOR SALE: Interdata (Perkin-Elmer) 7/16 Mini with 32KB core, front panel, 50A PWR supply. Includes HS tape reader, interfaces for LP, 2 (TTY), and RS-232 (Full duplex, programmable). Includes manuals and much SW (Basic, Fortran, OS, etc.,). **$800** - After 6 P 603 2035

COMPUTER AUTOMATION ALPHA 16; 16 k-word core memory,RTC PF-R. Modified Mod. ASR-33 TTY Manuals, utilities, assemblers and many option boards - 16 bit I/O Driver, 16 bit I/O, Asynch modem contr. 64 bit output, 10 bit A/D - D/A. Fairly complete documentation. Up and running in Fortran. Not much more than TTY at $1000. **Herb Sauer, 303-494-8724.**

FOR SALE: Heath H9 video terminal, excellent condition, $175 or best offer. You ship. [**214**] **962-4484**

WANTED: DIGITAL Group 32K memory board without memory chips and Phi deck controller board (kit, assembled or not working). W 1510 NW 35th. T 3

PET COMPUTERS moving up to LSI-11. Pet business system priced to sell. PET 2001-16N Computer $800; 2040 Dual Floppy 340K (holds more data than 6 TRS-80 disks) $1,100. Digital cassettes (2) $60 each. System complete with Text Editor, disk sort, database software, real estate software and more $2,100. Call **PAUL** (313)971-8447



COMPUTER SHOPPER, the new buy, sell, and trade publication, is ready to help you with the latest information on personal, small business and large-system computers, accessories and software.

Each ad-packed issue is full of bargains you are looking for. Included are ads from individuals throughout the United States who are selling their good, pre-owned equipment just so they can trade-up to new equipment coming on the market.

But, COMPUTER SHOPPER'S bargains won't be yours unless you subscribe. This useful, money-saving publication can become your way to communicate with other buyers, sellers, and traders all over the nation.

Whether you are a hobbyist or a part-time user, COMPUTER SHOPPER will put you in touch with the nationwide computer marketplace in time for you to take advantage of bargain opportunities.

Have something to sell? A COMPUTER SHOPPER subscriber probably wants to buy it.

Looking for a part, component or even a complete system? A COMPUTER SHOPPER subscriber probably wants to sell it.

COMPUTER SHOPPER is THE marketplace for anything in computers and is read by thousands of people who are ready to buy.

COMPUTER SHOPPER offers a unique format in which classified ads are categorized for fast location of specific items. Combining this with low individual ad rates — 10 cents a word — makes it the ideal place for buyers and sellers to communicate. And, its mix of individual, dealer, and manufacturer ads enable subscribers to find what they want at the best price possible.

COMPUTER SHOPPER will work for you in other ways, too. If you are just thinking about getting into computers, it can help you learn product availability and prices before you make a decision. And, through the timely ads, COMPUTER SHOPPER will keep you abreast of changes in the market which could create bargain opportunities for you.

BUT COMPUTER SHOPPER cannot work for you unless you subscribe.

Want to look us over first? We'll give you your first issue FREE and then bill you for the next 12. If you are not convinced COMPUTER SHOPPER suits your needs, just write "cancel" on the invoice and return it.

And, to let COMPUTER SHOPPER start working for you right now, with a paid subscription we'll also give you a FREE classified ad to sell your pre-owned equipment or to find equipment you want.

If you don't need to use the free classified ad now, use it anytime during your subscription.

Subscription: $10/year, 12 issues plus your first free one. Bank cards accepted. Money back guarantee.

The Nationwide Marketplace for Computer Equipment

COMPUTER SHOPPER

P.O. Box F22 • Titusville, Florida 32780
Telephone 305-269-3211

MICRO through the Ages

Since this issue marks the beginning of Volume 4 of **MICRO**, I thought it might be a good time to review the history of the magazine for any readers who might be interested.

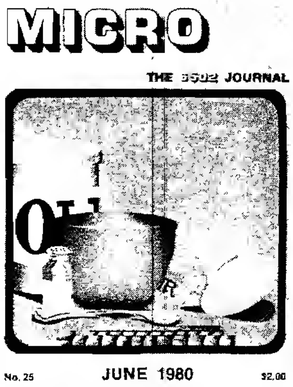
MICRO was first published in October 1977. The purpose of the journal was two-fold:
to promote the 6502 microprocessor and to provide an economical advertising medium for the 6502 world. In 1977, the 6502 was getting very little coverage in the major computer publications. An entire issue of **BYTE** or **Kilobaud** might contain, if you were lucky, one article relating to the 6502! As a KIM-1 enthusiast, I felt that this was unfair. I had started making products to support the KIM in November 1976 on a full-time basis. By late summer 1977 I had a couple of software packages, a power supply, and was in the advanced stages of development on a memory expansion board. I could see a problem arising: How to advertise these products. I could not afford to run ads in the national general purpose magazines for two reasons: first, the absolute cost was prohibitive; and, second, only a small percentage of the total readership would be interested in these KIM related products - making the cost of reaching potential customers very high. Thinking about these two problems, I decided the best remedy would be a high quality 6502 based journal which could serve to promote the 6502 in general, and to serve the 6502 based companies as an economical advertising medium. My initial expectation was that if I could include my own ads and break even on the publication, that would be doing okay. Our first issue was 28 pages long, printed at a 'store-front' print shop, and distributed to 450 subscribers and dealers shop, and 450 copies were distributed to subscribers and dealers. It was a start.

By October 1978, **MICRO** had grown in size, quality, and circulation. And, to my surprize, profitable! We were running about 52 pages, had changed printers twice as we outgrew them, and our readership was about 3000. The magazine was still published bi-monthly.

In February 1979, **MICRO** went monthly. This provided quicker turnaround for advertisers, and also increased the amount of material we could print. In May 1979, a separate corporation was formed for the purpose of publishing **MICRO** and other 6502 related material. Until this time, the magazine had been published by **The Computerist, Inc.**, my company which had continued to develop products for the KIM, and now the AIM and SYM as well. We felt that **MICRO** was conceptually and functionally a separate entity, with its own staff and equipment. **MICRO INK, Inc.**, was incorporated in May 1979. By this time we had outgrown another printer and moved on to Wellesley Press Inc., and the format we have today - high quality printing on glossy stock. We also acquired our own typesetting equipment during the summer. The circulation was about 6000 and each issue was 52 pages or more.

The changes in **MICRO** in the past year have been more evolutionary than revolutionary. It has expanded in size to 84 pages per issue, doubled in circulation to over 12,000, added new features, and now, with this issue, goes to full color covers. This was done primarily to allow those advertisers who will only advertise in color, to advertise with us. Secondarily, we hope that the colorful cover will be attractive both to our subscribers and those who purchase **MICRO** at their local computer store. We have ordered an option for our typesetter which will, in the near future, allow listings to be generated on any one of our microcomputers, and then sent directly to the typesetter. This will improve the quality and accuracy of the listings.

We would like to thank all of you who have helped **MICRO** grow and prosper, especially those of you who have written the articles which form the backbone of the publication. We look forward to an exciting new year serving the 6502 world.



MICRO in the Kitchen

Cover Artist
Terry Spillane

The Compleat Menu Planner

This month's cover depicts one of the ways in which the microcomputer may eventually aid the average family in planning menus in a systematic, dietically correct manner. I remember reading about this concept at least ten years ago, way back when the idea of a computer in the home was mind-boggling.

The Menu Planner assists the menu selection in several ways:

First, it helps to select each meal. As the user selects each item from a displayed list, new, related lists are presented. For example, the initial selection of **MEAT** *would result in a list of various types of meat. A selection of* **CHICKEN** *would cause a list of major ways to prepare chicken to appear: bake, broil, stew, etc. The entire, appetizing meal would quickly evolve.*

Second, the program would calculate the overall food value of the meal. It would have a list of the dietary requirements of the various members of the family, and would determine if these requirements were being met. It could make suggestions for changing particular items which were causing a dietary imbalance. The user could, presumably, override any such suggestions!

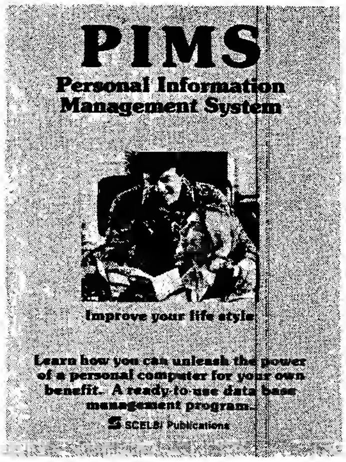
Third, as the meals for a period of time were determined, a shopping list could be generated which would take into account the meals for the week and the inventory on-hand.

Fourth, provided with a list of current prices for the items at the local supermarkets, it could determine where the individual products should be purchased, even taking into account the cost of getting to each store and personal preferences of the user.

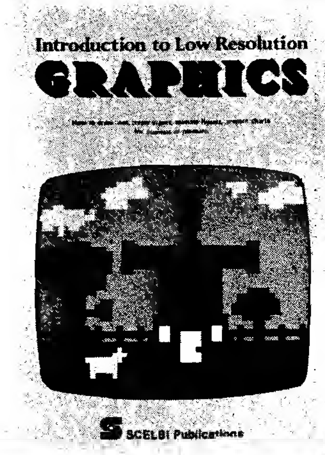
Fifth, when it comes time to prepare a particular meal, the recipies and other instructions are displayed. Cooking times are measured automatically, and, if desired, sensors are connected to measure the internal temperature of the meat, or whatever.

The capabilities are here now. Where are the programs? A rather inexpensive system should be able to perform all of the above functions, and earn its keep very quickly. If it could save $10 per week by finding the best buys, calculating correct amounts to purchase without waste, and keeping an eye on the inventory to prevent spoilage, that would be $500 per year!

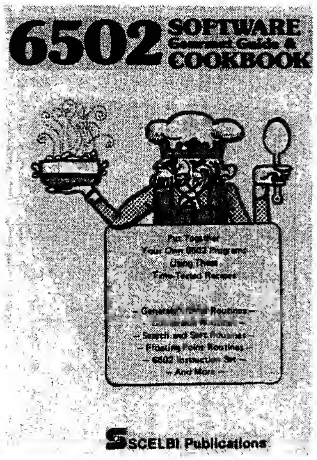
SERIOUS READING


PIMS
Personal Information Management System
Improve your life style
Learn how you can unleash the power of a personal computer for your own benefit. A ready-to-use data base management program.
SCELBI Publications

For business and for pleasure. Increase your personal capabilities . . . save money . . . improve your ability to plan . . . locate important facts quickly. This is a ready-to-use information (data base) management system on diskette for the Apple II. Plain-talk manual shows step-by-step how you can put this powerful program to work for you! Manage tax deductions, department store charges, simple accounts receivable and more! No. 91 *Just $24.95*



Now you can produce amazing computer graphics — even if you can't draw a straight line. Literally! Learn how to draw lines and shapes, make graphs, draw pictures and even do animations. The simple secrets of how to do all this are contained in this new book by Nat Wadsworth. Accompanying diskette contains all the programs and routines ready to run on your Apple II. You won't want to be without this fantastic capability. No. 92 *Just $24.95*



Program in machine language without tears! This latest cookbook by Robert Findley will show you how. You'll be able to put together programs without having to start from scratch. You'll have the most useful routines at your command — already programmed and ready to use with your Apple's 6502 microprocessor. Search and sort routines. Many general purpose utility routines. Floating point routines. No. 99 *Only $10.95*

SECRET GUIDE TO COMPUTERS
QUICKIE COURSE ON COMPUTERS
· Become a computer expert
· Learn how to do it in 20 minutes
· Discover BASIC the easy way
· Log your own "secret" programs
SCELBI Publications

Quickie course on computers! This book by Russ Walter will turn you into a computer expert quickly and easily. It explains the kind of computer found in most schools, small businesses and homes: the kind that has interactive BASIC. You'll learn BASIC, having fun every step of the way. Explains how to deal with computer machinery, which buttons to press and trains you to write many kinds of programs. No. 93 *Only $5.95*

● Books and programs for the discriminating Apple owner. . .

See them at your favorite computer or electronics store or order by mail with the coupon on right.

SCELBI Publications
20 Hurlbut Street, Elmwood, CT 06110

IMPORTANT ORDERING INFO! Include $1.00 shipping/handling charges for each item. Prices shown are for North American customers. Master Charge, VISA, Postal and Bank Money Orders preferred. Allow 4 weeks for delivery.

□ No. 91 □ No. 92 □ No. 93 □ No. 99
□ Free Catalog

Name (please print) ______
Card No. ______
Bank No. ______ Exp. Date ______
Address ______
City/State ______ Zip ______
Signature ______ Date ______

A Little Plus for Your APPLE II

For those of you who own an Apple II and are envious of the newer Apple II Plus, EDITPLUS provides you with some of the new features, at no cost!

Craig Peterson

A while back, Apple Computer, Inc., came out with a new version of their Apple II computer called the Apple II Plus. In this new machine comes the now famous Auto-Start ROM, and one of its neat features is a very much improved editing capability. In particular, for the standard Apple II owner, the 'non-copy' movement of the cursor requires two keystrokes for each column or row moved. (e.g. 'ESC', 'D', 'ESC', 'D', etc., etc.) Very tedious, and sometimes it is a bit unreliable.

On an Apple II Plus you just press 'ESC' and then use the I, J, K, or M key for cursor control up, left, right, and down respectively. And for really great action, you can use the repeat key along with the IJKM to speed the cursor 'non-copying' to any position on the screen. To get out of this editing mode, you just press any key other than the I, J, K or M key. This last key will be handled like a normal escape function and then you will be out of the special editor. Really nice, huh?

A second feature of the Apple II Plus is the ability to stop program listings. By pressing 'CTRL' 'S' during a listing, that blur of characters will be stopped so you can read the program. Pressing any key will begin the listing again right where it stopped. This works in both Integer Basic and Applesoft. It even works in the Monitor to stop a trace if you wish. In Applesoft, if the second key pressed is a 'CTRL' 'C', the listing will be aborted — just as you would expect normally.

Well, if you would like to be able to do this on your standard Apple II, you can either purchase the Auto-Start ROM, which has this and other features, for $65, or you can use the EDITPLUS program.

The EDITPLUS program is not very large and the way it works is fairly simple. Typing 'Call 768' revises the input and output hooks so that any I/O will be sent through EDITPLUS. The editing portion of the program, though the input hook, just looks for an 'ESC' character. If found, the program then checks the next character to see if it is an I, J, K, or M. If it is, the proper cursor action is performed and the next character is checked to see if it is an I, J, K, or M, and so on. The first non-IJKM character causes the program to do a normal escape function and then exit this mode. To totally disengage from this feature of EDITPLUS, just type 'IN#0', which restores the normal input hook address.

The control S feature of EDITPLUS uses the output hook. During any output, the program checks the keyboard strobe and if a 'CTRL' 'S' has been pressed, the output is stopped after the next carriage return. The EDITPLUS waits until a key is pressed again and at that time the output continues. If the second key is a 'CTRL' 'C', the keyboard strobe is left on so that Applesoft will see the 'CTRL' 'C' and abort the listing. To totally disengage from this feature of EDITPLUS, just type 'PR#0', which restores the normal output hook address.

An additional feature which I've added to all of this is escape L. By typing 'ESC' 'L', you leave whatever Basic you are in and jump to the Monitor, which is much quicker and easier than typing Call-151 all the time.

The assembly program listing for EDITPLUS is fairly self-explanatory.

The assembly program listing for EDITPLUS is fairly self-explanatory.

This example is assembled at good old page 3, hex address $300, but it could be anywhere you want. Also, this example is set up for use with 3.2 DOS on a 48K system. If you have 3.1 DOS and 48K memory, use DOS addresses $A7AD and $A99E in place of $A851 and $AA5B in lines 200, 210, 400, 640, and 690. If you have less than 48K, adjust these addresses downward a commensurate amount. Also, 3.1 DOS is peculiar in that it won't allow you to BRUN EDITPLUS right off the disk. You must BLOAD it, and then Call 768. If you don't have a disk system, simply change line 400 to RTS and delete lines 640, 680, and 690. If this change is made, it will be necessary to reassemble the program, or pad the revised lines with NOPs ($EA), because the branch addresses will change.

So there you have it. A nice edit program for your Apple II. No longer do you need to be jealous of those folks that have an Apple II Plus. You too can have fun editing (and TRACE and STEP too, heh! heh!).

µ

Craig Peterson is an engineer who has owned an Apple II since May of 1979. Since then, he has enjoyed working with it to such an extent, that he recently persuaded his employer to buy one to aid in programming numerical control routing machines.

```
                :********************************
                :*          EDITPLUS            *
                :*              BY              *
                :*       CRAIG PETERSON         *
                :*        DECEMBER 1979         *
                :*                              *
                :*A PROGRAM TO GIVE THE STANDARD*
                :* APPLE II THE ENHANCED CURSOR *
                :* EDITING CAPABILITIES OF THE  *
                :*       APPLE II PLUS          *
                :********************************
                :
                CH   .DL 0024
                BASL .DL 0028
                YSAV .DL 0035
                CSWL .DL 0036
                CSWH .DL 0037
                KSWL .DL 0038
                KSWH .DL 0039
                DOS  .DL A851
                YDOS .DL AA5B
                KBRD .DL C000
                STRB .DL C010
                ESC1 .DL FC2C
                RKEY .DL FD0C
                OUT1 .DL FDF0
                KEYN .DL FD1B
                MNTR .DL FF65
                :
                     .OR 0300
                :
0300   AD7E03   BGIN LDA KADR     CHANGE INPUT &
0303   8538          STA *KSWL     OUTPUT POINTRS
0305   AD7F03        LDA KADR+01   TO NEW ROUTINE
0308   8539          STA *KSWH     AT 'SKEY' AND
030A   AD8003        LDA VADR      'SVID' RESP.
030D   8536          STA *CSWL
030F   AD8103        LDA VADR+01
0312   8537          STA *CSWH
0314   4C51A8        JMP DOS      CHG DOSPTRS&RTN
0317   201BFD   SKEY JSR KEYN     GET NEXT CHAR
031A   C99B          CMP 9B       IS CHAR='ESC'?
031C   F00B          BEQ ESC2     IF SO, GO ESC2
031E   60            RTS          IF NOT, RETURN
031F   38       SPCL SEC          PREPARE A PTR &
0320   E9C9          SBC 0C9       TURN I,J,K,M
0322   A8            TAY           INTO D,B,A,C
0323   B97903        LDA TABL,Y    RESPECTIVELY
0326   202CFC        JSR ESC1     DO STANDARD ESC
0329   A424     ESC2 LDY *CH      GET THE NEXT
032B   B128          LDA (BASL),Y  INPUT CHARACTF
032D   48            PHA
032E   293F          AND 3F
0330   0940          ORA 40
0332   9128          STA (BASL),Y
0334   68            PLA
0335   201BFD        JSR KEYN
0338   C9CE          CMP 0CE      IS CHAR>='N'?
033A   B00E          BCS RTRN      THEN RETURN
033C   C9C9          CMP 0C9      IS CHAR<'I'?
033E   900A          BCC RTRN      THEN RETURN
0340   C9CC          CMP 0CC      IS CHAR='L'?
0342   D0DB          BNE SPCL     IF<>'L',DO SPCL
```

```
0344   2051A8   0640          JSR DOS        IF=L, RESET DOS
0347   4C65FF   0650          JMP MNTR        PNTRS & ->MNTR
034A   38       0660   RTRN   SEC            ITS NOT I,J,K,M
034B   202CFC   0670          JSR ESC1        SO DO STD ESC
034E   A424     0680          LDY *CH        CORRECT YSAVE
0350   8C5BAA   0690          STY YDOS        REG IN DOS
0353   4C0CFD   0700          JMP RKEY        AND RETURN
0356   8435     0710   SVID   STY *YSAV      SAVE Y
0358   C98D     0720          CMP 8D         IS CHAR = CR?
035A   D018     0730          BNE RETN       IF NOT, RETURN
035C   AC00C0   0740          LDY KBRD       GET KBRD CHAR
035F   1013     0750          BPL RETN       NO STROB, RETRN
0361   C093     0760          CPY 93         IS IT CTRL 'S'?
0363   D00F     0770          BNE RETN       IF NOT, RETURN
0365   2C10C0   0780          BIT STRB       CLEAR KEY STROB
0368   AC00C0   0790   AGIN   LDY KBRD       IS KEY PRESSED?
036B   10FB     0800          BPL AGIN       IF NOT, TRY AGN
036D   C083     0810          CPY 83         IS IT CTRL 'C'?
036F   F003     0820          BEQ RETN       IF SO,LEAV STRB
0371   2C10C0   0830          BIT STRB       CLEAR KEY STROB
0374   A435     0840   RETN   LDY *YSAV      RESTORE Y &
0376   4CF0FD   0850          JMP OUT1        REJOIN OUTPUT
0379   C4C2C1   0860   TABL   .HS C4C2C1FFC3
037C   FFC3
037E   1703     0870   KADR   .SA SKEY       STOR 'SKEY' ADR
0380   5603     0880   VADR   .SA SVID       STOR 'SVID' ADR
                0890          .EN
```

Skyles Electric Works

Presenting the Skyles MacroTeA

The Software Development System For the Serious Programmer

Text Editor

To help you write your program, MacroTeA includes a powerful text editor with **34 command functions**:

AUTO Numbers lines automatically.

NUMBER Automatically renumbers lines.

FORMAT Outputs text file in easy-to-read columns.

COPY Copies a line or group of lines to a new location.

MOVE Moves a line or group of lines to a new location.

DELETE Deletes a line or group of lines.

CLEAR Clears the text file.

PRINT Prints a line or group of lines to the PET screen.

PUT Saves a line or group of lines of text on the tape (or disc).

GET Loads a previously saved line or group of lines of text from the tape (or disc).

OUPLICATE Copies text file modules from one tape recorder to the other. Stops on specific modules to allow changes before it is duplicated. This command makes an unlimited length program (text file) practical.

HARD Prints out text file on printer.

ASSEMBLE Assembles text file with or without a listing. Assembly may be specified for the object code (program) to be recorded or placed in RAM memory.

PASS Does second pass of assembly. Another command that makes unlimited length text files (source code) practical.

RUN Runs (executes) a previously assembled program.

SYMBOLS Prints out the symbol table (label file).

SET Gives complete control of the size and location of the text file (source file), label file (symbol table) and relocatable buffer.

DISK Gives complete access to the eleven DOS commands:
PUT GET NEW INITIALIZE
DIRECTORY COPY DUPLICATE
SCRATCH VALIDATE RENAME
ERROR REPORT

EDIT Offers unbelievably powerful search and replace capability. Many large computer assemblers lack this sophistication.

FIND Searches text file for defined strings. Optionally prints them and counts them; i.e., this command counts number of characters in text file.

MANUSCRIPT Eliminates line numbers on PRINT and HARD command. Makes MacroTea a true and powerful Text Editor.

BREAK Breaks to the Monitor portion of MacroTea. A return to Text Editor without loss of text is possible.

USER Improves or tailors MacroTea's Text Editor to user's needs; "Do-it-yourself" command.

Fast...Fast Assembler

Briefly, the pseudo-ops are:

- **BA** Commands the assembler to begin placing assembled code where indicated.
- **CE** Commands the assembler to continue assembly unless certain serious errors occur. All errors are printed out.
- **LS** Commands the assembler to start listing source (text file) from this point on.
- **LC** Commands the assembler to stop list source (text file) from this point in the program.
- **CT** Commands the assembler to continue that source program (text file) on tape.
- **OS** Commands the assembler to store the object code in memory.
- **OC** Commands the assembler to not store object code in memory.
- **MC** Commands the assembler to store object code at location different from the location in which it is assembling object code.
- **SE** Commands the assembler to store an external address.
- **DS** Commands the assembler to set aside a block of storage.
- **BY** Commands the assembler to store data.
- **SI** Commands the assembler to stora an internal address.
- **DE** Commands the assembler to calculate an external label expression.
- **DI** Commands the assembler to calculate en internal label expression.
- **EN** Informs the assembler that this is the end of the program.
- **EJ** Commands the assembler to eject to top of page on printer copy.
- **SET** A directive not a pseudo-op, directs the assemblers to redefine the value of a label.

Macro Assembler

The macro pseudo-ops include:

MD This is a macro beginning instruction definition.

ME This is end of a macro instruction definition.

EC Do not output macro-generated code in source listing.

ES Do output macro-generatad code in source listing.

Conditional Assembler

The conditional assembly pseudo-ops are:

IEQ If the label expression is equal to zero, assemble this block of source code (text file).

INE If the label expression is not equal to zero, assemble this block of source code (taxt file).

IPL If the label expression is positive, assemble this block of source code.

IMI If the label expression is negative, assemble this block of source code.

******* This is the end of a block of source code.

Enhanced Monitor

. . . By having 16 powerful commands:

A Automatic MacroTeA cold start from Monitor.

Z Automatic MacroTeA warm start from Monitor.

F Loads from tape object code program.

S Saves to tape object code between locations specified. .

D Disassembles object code back to source listing.

M Displays in memory object code starting at selected location. The normal PET screen edit may be used to change the object code.

R Displays in register. Contents may be changed using PET screen edit capabilities.

H Hunts memory for a particular group of object codes.

W Allows you to walk through the program one step at a time.

B Breakpoint to occur after specified number of passes past specified address.

Q Start on specified address. Quit if STOP key or breakpoint occurs.

T Transfers a program or part of a program from one memory area to another.

G Go!! Runs machine language program starting at selected location.

X Exits back to BASIC.

I Display memory and decoded ASCII characters.

P Pack (fill) memory with specified byte.

What are the other unique features of the MacroTeA?

- Labels up to 10 characters in length
- 50 different symbols to choose from for each character
- 10^{16} different labels possible
- Create executable object code in memory or store on tape
- Text editor may be used for composing letters, manuscripts, etc.
- Text may be loaded and stored from tape or disc
- Powerful two-cassette duplicator function
- String search capability
- Macros may be nested 32 deep
- 25 Assembler psuedo-ops
- 5 Conditional assembler psuedo-ops
- 40 Error codes to pinpoint problems
- 16 Error codes related to Macros
- Warm-start button
- Enhanced monitor with 16 commands

Truly, there is simply no other system of this magnitude at anywhere near this price. **$395.00***

(With any Skyles Memory Expansion System, **$375.00**

California residents: please add 6% or 6 5% sales tax as required

VISA, MASTERCHARGE ORDERS CALL **(800) 538-3083** (except California residents)
CALIFORNIA ORDERS PLEASE CALL (408) 257-9140

Skyles Electric Works

231 E South Whisman Road
Mountain View, CA 94041
(415) 965-1735

Lower Case Lister

A 'bug' in the PET/CBM model 2022 and 2023 printers made before February 1980 causes listings to be printed as graphic symbols. This program provides a remedy.

Rev. James Strasma

One of the best features of the PET/CBM computers is their ability to easily use both upper and lower case letters. This same capability is included in the CBM model 2022 and 2023 printers. Unfortunately, printers made before February, 1980 cannot print in the lower and upper case mode unless a control character is sent at the start of EVERY line that needs lower case. This means that program printouts can be made to look very much like ordinary typed output. Listings, however, all come out in graphics mode. Graphics characters are substituted for all the upper-case characters in the listing, and all the lowe-case characters come out in upper-case. (See the first sample Basic listing for an example of this.)

This leaves the programmer with three choices: 1) stick to graphics mode entirely, 2) learn to translate graphics into alpha, or 3) find a new way to list programs. I made the third choice, and lower-case lister is the result. It emulates the list routine in the CBM/PET Basic ROMs closely. Essentially, the routine prints the all-important (cursor down) character at the beginning of each line in the listing. This tells the printer to treat the rest of the line as lower-case. One other major change was necessary. At present, the PET printer errs in printing about 20 of the characters when in the lower-case mode. It replaces the correct character with the one having the opposite value in the high bit. This ups or cuts the character's value by 128. The correction involved screening each character before it is output, and flipping the high bit back again. One added consideration is that this is not to be done if the character involved is part of a Basic keyword. Those are ouput without being screened. Note that the problem is in the printer, not the computer. Thus, if the 'fixed' characters are directed to the screen instead of a printer, they will be incorrect there. The result of the fix is that a listing made with the lower-case lister will look exactly like the same program would if listed normally in lower-case mode on the screen.

Because this is the sort of program I need all the time, I squeezed it into the second cassette buffer, an area safe not only from Basic, but even safe from hardware resets. Only a power failure disturbs that buffer. The lister is short enough not to interfere with the ROM monitor or the Basic Programmer's Tool kit, both of which use parts of the buffer. Two prices are paid for the choice location and compatibility. First, the only option is to list the entire program. Second, very long lines that extend onto a third line of the screen or second line of the printout when listed will revert to 'normal' faulty printing at the beginning of the extra line. Fixes for both of these are possible, but not in the space available.

For those of you with 'old ROMs', the program will need some changes. Nearly all the external references of the program (lines 190-290 in the assembler source) will have to be changed, along with the resulting object code. The changes for the assembler source code in the listing are given in Table 1. The object code changes are given in Table 2. With these changes, the program does work with the old ROM PETs.

Once you have typed in the correct code for your machine, making a lower-case lisiting is easy. The commands are almost the same as usual:

open 4,4 to wake up the printer

print#4, a home character to set the paging mode

omd 4 to make the printer the output device

sys(826) instead of list to make the listing

print#4 to return command to the terminal

I'm told that eventually you will be able to buy a retrofit ROM to clean up lower-case listings automatically, at a cost of about $40. Now it won't be necessary to wait or to pay that price. Which would you rather have, cryptic Basic listings like the first sample program here, or clean listings like the one below it? the choice is yours.

μ

James Strasma has been a United Methodist pastor for five years. He learned programming from books, and he has owned a PET for two years. He initially became interested in programming to do some work for the church.

Aside from his duties as a pastor, and the writing he does for Micro, Rev. Strasma also occupies his time organizing a users' group. This group is interested in C.W. Moser's assembler TED, in different 6502 versions.

END OF MAE PASS!

```
                        ;Lower-Case Lister for CBM Printers

                        ; Cleans up PET program listings


                        ;  by Rev. James Strasma
                        ;      120 W. King Street
                        ;      Decatur, Il. 62521

                        ;     as of Feb. 29, 1980


                                .ba $033a           ;sys(826)
                        ;     (2nd. cassette buffer)


                        ;Label values given for new ROMs

                        crlf    .de $c9e2          ;Return+line feed
                        findlin .de $c52c          ;Find Basic line
                        linstor .de $11            ;stores line #
                        memory  .de $46            ;Temporary storage
                        prchar  .de $ca45          ;Prints a character
                        print   .de $dcd9          ;Print integer value
                        quotflg .de $09            ;Flags Quote mode
                        ready   .de $c389          ;Basic warm start
                        toktabl .de $c092          ;Start of token table
                        tstop   .de $ffe1          ;Test stop key
                        where   .de $5c            ;Block transfer ptr. #2
033A- A9 00                     lda ##00           ;First line # is 0
033C- 85 11                   sta *linstor       ;Store it
033E- 85 12                     sta *linstor+1
0340- 20 2C C5                  jsr findlin        ;Find address of line #0
0343- 68                        pla                ;Ignore last 2
0344- 68                        pla                ;items in the stack
0345- A0 01             newline ldy #1             ;Set up char. counter
0347- 84 09                     sty *quotflg       ;Not in quote mode
0349- B1 5C                     lda (where),y      ;Hi byte of forward link
034B- F0 46                     beq done           ;0 here=end of text
034D- 20 E1 FF                  jsr tstop          ;Abort if stop key down
0350- 20 E2 C9                  jsr crlf           ;Else, start new line
0353- C8                        iny
0354- B1 5C                     lda (where),y      ;Lo byte of line #
0356- AA                        tax
0357- C8                        iny
0358- B1 5C                     lda (where),y      ;& hi byte
035A- C9 FF                     cmp ##ff           ;$ffff=maximum line #
035C- D0 04                     bne notlast        ;If less
035E- E0 FF                     cpx ##ff           ;Now check low byte
0360- F0 31                     beq done           ;If line # is $ffff
0362- 84 46             notlast sty *memory        ;Store counter
0364- 20 D9 DC                  jsr print          ;Print integer line #
0367- A9 11                     lda ##11           ;Cursor down sets
0369- 20 45 CA                  jsr prchar         ;lower-case mode
036C- A9 20                     lda #'             ;Then send a space
036E- A4 46             chrmain ldy *memory        ;Remember counter value
0370- 29 FF                     and #%11111111
0372- 20 C2 03          send    jsr out            ;Send char. in r(a)
0375- C9 22                     cmp #'"            ;In quote mode?
0377- D0 06                     bne notquot        ;If not
```

```
0379- A5 09                   lda *quotflg      ;Remember counter value
037B- 49 FF                   eor #%11111111    ;Flip quote flag
037D- 85 09                   sta *quotflg      ;Store it again
037F- C8             notquot  iny               ;Up counter
0380- F0 11                   beq done          ;If it was $ff, quit
0382- B1 5C                   lda (where),y     ;Read next character
0384- D0 10                   bne notlend       ;0=end of line
0386- A8                      tay               ;Prepare to read
0387- B1 5C                   lda (where),y     ;forward link
0389- AA                      tax               ;for the start location
038A- C8                      iny               ;of the next line
038B- B1 5C                   lda (where),y     ;r(x)=lo, r(a)=hi
038D- 86 5C                   stx *where        ;Next line now the
038F- 85 5D                   sta *where+1      ;current line
0391- D0 B2                   bne newline       ;Unless forward link=0
0393- 4C 89 C3       done     jmp ready        ;Quit & goto Basic
0396- 10 DA          notlend  bpl send          ;Tokens are > $80
0398- C9 FF                   cmp ##ff          ;May be a token, check
039A- F0 D6                   beq send
039C- 24 09                   bit *quotflg      ;If in quote mode,
039E- 30 D2                   bmi send          ;can't be a token
03A0- 38                      sec
03A1- E9 7F                   sbc #%01111111    ;Convert from ASCII
03A3- AA                      tax               ;to token #
03A4- 84 46                   sty *memory       ;Store counter value
03A6- A0 FF                   ldy ##ff          ;set flag
03A8- CA             nextokn  dex               ;Look at next token
03A9- F0 08                   beq ritetkn       ;If found the right one
03AB- C8             nextchr  iny
03AC- B9 92 C0                lda toktabl,y    ;Next char. of token
03AF- 10 FA                   bpl nextchr       ;Last char. of token
03B1- 30 F5                   bmi nextokn       ;is in opposite case
03B3- C8             ritetkn  iny               ;Got correct token
03B4- B9 92 C0                lda toktabl,y     ;Read next char.
03B7- 30 05                   bmi fliptkn       ;Flip case if last
03B9- 20 45 CA                jsr prchar        ;Send token char.
03BC- D0 F5                   bne ritetkn       ;Jump
03BE- 49 80          fliptkn  eor #%10000000    ;Flip last char.
03C0- D0 AC                   bne chrmain       ;Jump
03C2- 48             out      pha               ;Store character
03C3- 09 C0                   ora #%11000000    ;Is this a char-
03C5- C9 DB                   cmp #%11011011    ;acter that CBM
03C7- 90 08                   bcc outrang       ;printers flub up
03C9- C9 E0                   cmp #%11100000    ;in lower-case?
03CB- 10 04                   bpl outrang       ;If not
03CD- 68                      pla               ;Remember character
03CE- 49 80                   eor #%10000000    ;Apply a correction
03D0- 48                      pha               ;Store it
03D1- 68             outrang  pla               ;Remember for printing
03D2- 4C 45 CA                jmp prchar        ;Use rts there
                 1120         .en
END OF MAE PASS!
--- LABEL FILE: ---

chrmain =036E         crlf =C9E2          done =0393
findlin =C52C         fliptkn =03BE        linstor =0011
memory =0046          newline =0345        nextchr =03AB
nextokn =03A8         notlast =0362        notlend =0396
notquot =037F         out =03C2            outrang =03D1
prchar =CA45          print =DCD9          quotflg =0009
ready =C389           ritetkn =03B3        send =0372
toktabl =C092         tstop =FFE1          where =005C
//0000,03D5,03D5
```

Sample 1 (Bad)

```
100 REM> SAMPLE BASIC PROGRAM
110 REM> BEFORE LOWER-CASE LISTER
120 REM>
130 PRINT"-IRST, SOME ╱TT‐ ¬♦♥‾
140 PRINT"MIXED WITH LOWER CASE.
150 PRINT"║HEN SOME NUMBERS,
160 PRINT"LIKE 1,2,3.14, & SUCH,
170 PRINT"BOTH IN & OUT OF QUOTES
180 A=1:B=2:C=3.14
190 PRINT"█HEN SOME CURSOR-CONTROL
200 PRINT"█CHARACTERS...████
210 PRINT"¬VEN GRAPHICS, LIKE '▄'
220 PRINT"-INALLY, THE ONES THAT
230 PRINT"THE -|\ PRINTERS DON'T
240 PRINT"LIKE IN LOWER-CASE MODE:
250 PRINT"← & ↑=LEFT & UP ARROWS,
260 PRINT"[]=BRACKETS & \/=SLASHES,
270 PRINT"█=TAB & █=ESCAPE.
280 PRINT"╱OTE '↑' IN THE EQUATION
290 E=MC↑2
300 END
READY.
```

Sample 2 (Good)

```
100 rem> sample basic program
110 rem> after lower-case lister
120 rem>
130 print"First, some UPPER CASE
140 print"mixed with lower case.
150 print"Then some numbers,
160 print"like 1,2,3.14, & such,
170 print"both in & out of quotes
180 a=1:b=2:c=3.14
190 print"█Then some cursor-control
200 print"█characters...████
210 print"Even graphics, like '▄'
220 print"Finally, the ones that
230 print"the CBM printers don't
240 print"like in lower-case mode:
250 print"← & ↑=left & up arrows,
260 print"[]=brackets & \/=slashes,
270 print"█=tab & █=escape.
280 print"Note '↑' in the equation
290 e=mc↑2
300 end
READY.
```

Table 1

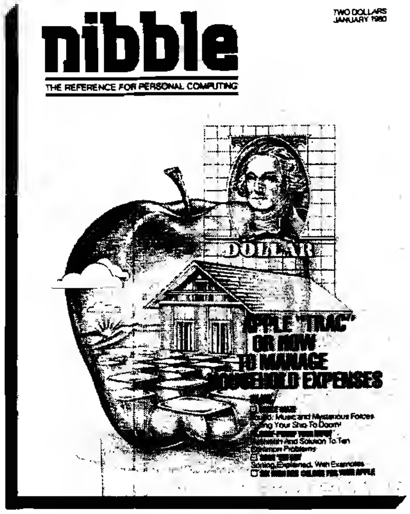
SOURCE CHANGES FOR OLD ROM PETS

```
190     crlf      .de     $c9d2     ;Return+line feed
200     findlin   .de     $c522     ;Find ascii line
210     linstor   .de     $08       ;Stores line #
220     memory    .de     $98       ;Temporary storage
230     prchar    .de     $ca49     ;Prints a character
240     print     .de     $dc9f     ;Print integer value
250     quotflg   .de     $60       ;Flags quote mode
260     ready     .de     $c38b     ;Basic warm start
290     where     .de     $ae       ;Block transfer ptr. #2

270 toktabl & 280 tstop remain unchanged
```

Table 2 **:Object Code Changes for Old ROM Pets**

```
Change $09 to $60 at:
         $0348
          037a
          037e
          039d

Change $11 to $08 at $033d

Change $12 to $09 at $033f

Change $2c to $22 at $0341

Change $45 to $49 at:
         $036a
          03ba
          03d3

Change $46 to $98 at:
         $0363
          036f
          03a5

Change $5c to $ae at:
         $034a
          0355
          0359
          0383
          0388
          038c
          038e
          0390

Change $89 to $8b at $0394

Change $d9 to $9f at $0365

Change $e2 to $d2 at $0351
```

INTRODUCING . . . NIBBLE THE REFERENCE FOR APPLE COMPUTING


nibble
THE REFERENCE FOR PERSONAL COMPUTING
TWO DOLLARS
JANUARY 1980
DOLLAR
APPLE "TRAC" OR HOW TO MANAGE HOUSEHOLD EXPENSES
Music and Mysterious Forces
Your Ship To Doom!
And Solution To Ten
Common Problems
Sorting Explained, With Examples

NIBBLE IS:
A SOFTWARE GUIDE for high quality Applications Programs for your Home and Business.

NIBBLE IS:
A REFERENCE GUIDE to new Programming Methods.

NIBBLE IS:
A BUYERS GUIDE for making purchase decisions on new products.

NIBBLE IS:
A CONSTRUCTION PROJECT COOKBOOK for adding function and value to the system you already own.

NIBBLE IS:
A COMMUNICATIONS CLEARING HOUSE for users, vendors, and associations.

Each issue of NIBBLE features at least one significant new application program of commercial quality. The programs in NIBBLE are surrounded with articles which show how to USE the programming methods in your OWN programs.

Examples of upcoming articles:
□ Modeling and Forecasting Your Business □ Build a Two-Tape Controller for $12
□ Arcade Shooting Gallery — Save Your Quarters! □ Data Base Management System I, II, III

And many many more! NIBBLE will literally "Nibble Away" at the mysteries of your system to help you USE IT MORE. In 1980, the principal featured system is the Apple II.

Try a NIBBLE

nibble
BOX 325 Lincoln, Mass. 01773

I'll try NIBBLE!
Enclosed is my $15 for 8 issues.
□ **check** □ **money order**

Name ______________________

Address ______________________

City ______________________

State ______________ Zip ________

Put Your Hooks Into OSI BASIC

Is it possible to extend your OSI BASIC-IN-ROM? Yes. In fact it is very easy, if you know how, and this article you will teach you how!

Edward H. Carlson

The OSI BASIC-IN-ROM has a remarkable feature which allows you to meddle with one of the innermost subroutines; the one that is used to parse a line of code character by character. It was intended to be tampered with, I am sure. Why else would the subroutine lie in page zero ($BC to $D3), copied there from its home in the BASIC ROM at $BCEE? Listing 1 gives this "character parsing" routine.

This Microsoft BASIC, Version 1.0 Rev. 3.2, is used on all non-disk Ohio Scientific machines. I expect that a similar routine lies in page zero of the other 6502 Microsoft BASICs, such as the one for PET, and has been exploited by other hackers to make extensions of BASIC for their machines. In Listing 2, I show one way of doing this for use on my C2-4P and OSI C1 and C2 machines in general.

There are some subtle points which arose during this programming task, and the best way to explain them is to remind you of some of the "basic" facts about OSI BASIC. It operates in two modes, "immediate" and "RUN". In the immediate mode, you can enter a line of code, preceded or not by a line number. This code is entered into a buffer starting at $13. When RETURN is hit, the first character of the line is picked up by the little parser which examines it to determine the fate of the line. If the line started with a numeral, then it is destined to be entered in the source code table. Otherwise it is executed

List 1

```
 10 0000          : ***** CHARACTER PARSER *****
 20 0000          :
 90 00BC           *      =$BC
100 00BC E6C3     P0      INC $C3    INCREMENT LO BYTE OF ADDRESS
110 00BE D002             BNE P1
120 00C0 E6C4             INC $C4    INCREMENT HI
130 00C2 ADFFFF   P1      LDA $FFFF  LOAD A WITH CHARACTER
140 00CS C93A             CMP #':    IS IT A COLON?
150 00C7 B00A             BCS P2     IF YES, BRANCH AND START NEW LINE
154 00C9 C920             CMP #'     IS IT A SPACE?
156 00CB F0EF             BEQ P0     IF YES, GET ANOTHER CHAR.
160 00CD 38               SEC        SET CARRY FLAG
170 00CE E930             SBC #$30   SUBTRACT $30
180 00D0 38               SEC        SET CARRY FLAG
190 00D1 E9D0             SBC #$D0   SETS C FLAG FOR ASCII NUMERAL
200 00D3 60       P2      RTS        END OF SUBROUTINE, CHAR. IN A
```

List 2

```
 10 0000          : ***** HOOK TO OSI BASIC *****
 15 0000          :
 96 0000          ; ***** INITIALIZATION ROUTINE *****
 97 0000          ;
 98 0000          : STORE 6 BYTES IN ZERO PAGE TO JUMP TO THIS EXTENDED
 99 0000          ; BASIC ROUTINE
100 0000          :
102 0000           LO     =$C3       LO BYTE OF CHAR. ADDRESS STORAGE
103 0000           SCREEN =$FE       LO BYTE OF SCREEN ADDRESS STORAGE
10S 0222           *      =$0222
110 0222 A94C             LDA #$4C
115 0224 8SBC             STA $BC
120 0226 A939             LDA #STARLO
12S 0228 85BD             STA $BD
130 022A A902             LDA #STARHI
13S 022C 8SBE             STA $BE
140 022E A9EA             LDA #$EA
142 0230 8SBF             STA $BF
143 0232 85C0             STA $C0
144 0234 85C1             STA $C1
150 0236 4C0000           JMP $0000  JUMP TO BASIC WARM START
190 0239          ;
191 0239          ; ***** MAIN ROUTINE *****
192 0239          ;
193 0239          ; CHECK THE BASIC SOURCE CODE LINE FOR THE CHARACTER
194 0239          : # OR %.
195 0239          ;
200 0239 E6C3     START   INC LO     INCREMENT LO BYTE OF ADDRESS
202 023B          STARHI =START/256
203 023B          ZZZ    =STARHI*256
204 023B          STARLO =START-ZZZ
```

immediately*in situ*. Supposing the line starts with a numeral. The parser examines subsequent characters, transfering the numerals to another routine, until it finds a non-numeral character. Then the parser quits, handing the task of tokenizing the line and storing it in the source code table to another routine.

Our strategy is to splice into the character parser subroutine with a jump, so as to take a look at the current character before BASIC gets its hands on it. I picked two characters, % and #, not used by BASIC, to signal that we intend to message this line ourselves. We need two such labels because we first must safely transfer the line from the buffer to the source code table without triggering any special fireworks, yet on subsequent encounters with the line (during RUN) the spliced code in the parser must take special actions.

Listing 3 shows how our special line of code looks in a BASIC program. The line number is followed immediatley by the # symbol, which is followed by a letter (or other symbol) and then by a null (not visible on the screen, but used in the source code table as a line terminator) or a colon (line continues with a new, independent statement of code). Now the sticker is that when inputing the line from the keyboard, we do not type "10#C" for example, but "10%C". When return is hit, the % sign triggers (at line 287 of listing 2) a replacement of % by # in the line of code in the buffer. Then the "tokenizer" moves the code, now reading "10#C", to its spot in the source code area of memory. When RUN is hit and execution reaches the # in the "10#C", our spliced code at line 282 branches to line 400 which calls the parser again. This is an example of reentrant coding since we were already in a (spliced in) section of the same subroutine! The extra call to the parser, which picks up the character after the #, means that BASIC outside our splice never sees the # symbol. This is essential because BASIC would have to fit over the # and would exclaim "syntax ERROR" and break. (I know. I got quite a few of them before I devised this somewhat cumbersome %, # trick.)

I show only one extension to BASIC, the rapid screen clear which is useful during games. The screen clear is signaled by the C after the #. One can put any number of CMP, BEQ pairs after line 420, one for each extension subroutine. I have a "rapid" tape read-write routine (three times as fast as OSI's) which I intend to implement from BASIC. Other possibilities include a built in hex-to-decimal conversion so I can write "10#H Q=D000" instead of "10 Q=53248", which I find hard to remember.

Notice that the clear subroutine ends with a jump to the beginning of

List 2 continued

```
210 023B D002               BNE 51
220 023D E6C4               INC LO+1   INCREMENT HI
230 023F A5C3        51     LDA $C3    STORE CHARACTER ADDRESS
240 0241 8D4A02             STA ADD+1
250 0244 A5C4               LDA $C4
260 0246 8D4B02             STA ADD+2
270 0249 ADFFFF      ADD    LDA $FFFF  A CONTAINS THE CHARACTER
271 024C            ; FFFF IS A DUMMY ADDRESS, REAL FROM $00C3,C4
280 024C C923               CMP #'#    IS IT # ?
282 024E F00D               BEQ EXTEND IF YES, BRANCH TO SUBROUTINE
284 0250 C925               CMP #'%    IS IT % ?
285 0252 D006               BNE E1
287 0254 A000               LDY #0     IF YES, CHANGE % TO #
288 0256 A923               LDA #'#
289 0258 91C3               STA ($C3),Y
299 025A 4CC200      E1     JMP $C2    BACK TO PARSING THE BASIC LINE
400 025D 20BC00      EXTEND JSR $00BC  TEST FOR WHICH SUBROUTINE
410 0260 C943               CMP #'C
420 0262 F003               BEQ CLRSCR
450 0264 4C5C00             JMP $005C  BACK TO BASIC PARSING THE LINE
490 0267            ;
491 0267            ; ***** CLEAR SCREEN ROUTINE *****
492 0267            ;
493 0267            ; THE SCREEN CONSISTS OF 8 PAGES STARTING AT $D000
494 0267            ;
500 0267 A200        CLRSCR LDX #0
505 0269 86FE               STX SCREEN
506 026B A9D0               LDA #$D0
507 026D 85FF               STA SCREEN+1
510 026F A000               LDY #0
520 0271 A920               LDA #$20
530 0273 91FE        C2     STA (SCREEN),Y
540 0275 C8                 INY
545 0276 C000               CPY #0
550 0278 D0F9               BNE C2
552 027A E6FF               INC SCREEN+1
555 027C E8                 INX
556 027D E008               CPX #8
560 027F D0F2               BNE C2
565 0281 4CBC00             JMP $00BC
570 0284            ; JMP TO GET NEXT CHARACTER, A NULL, AND SO BASIC
571 0284            ; SEES THE END OF THE LINE OF SOURCE CODE AND GOES
572 0284            ; TO THE NEXT LINE.
```

List 3

```
 10 REM BASIC PROGRAM TO ILLUSTRATE THE USE OF THE EXTENDED
 11 REM  FEATURE OF THE BASIC: THE RAPID SCREEN CLEAR.
 15 PRINT"HI"
 20 #C
 30 PRINT"IS THE CHARACTER % A PROBLEM?"
 35 PRINT"IS THE CHARACTER # A PROBLEM?"
 40 PRINT"HERE IT COMES!":#C:PRINT"THERE IT WAS!"
 50 5#C
 60 PRINT"THAT ONE DIDN'T WORK"
 70 REM THE "#" IN THE LINES 20,40 AND 50 WERE ENTERED FROM
 71 REM  THE KEYBOARD AS A "%".
OK
```

the parser, incrementing the character pointer again. We must drop the C as well as the # because the line "10C" would trigger a syntax ERROR break. So our line, which started as "10%C", finally looks to be just a naked "10" to BASIC, which it shrugs off as legal, but void of purpose.

When trouble shooting this program, it is very helpful to put a STA to screen just after line 270 so you can see what character is being messaged. In fact, I built a crude character-by-character trace routine by adding a call to the keyboard routine at $FC00 so that each character remained on the screen to be puzzled over, until I hit a key and went on to the next. An elegant line-by-line trace routine could be built, triggered by the null at the end of each line of stored source code.

The line "#C", *sans* line number, executes o.k. in the immediate mode. One need not avoid the # or % symbols when placed in strings, as Listing 3 illustrates. Apparently the character parser never penetrates inside the quotes of a string. On the other hand, the line "50 S#C" in Listing 3 looks like "10 S" to BASIC and causes an ERROR break.

Honesty now compels me to mention the price to be paid for extending BASIC. It stands at a computed 36 microseconds per character examined. This could be reduced somewhat by tightening up the code. This is the full price (in time units) because the extension subroutines, no matter how many, cost no time until they are called by the letter following the # symbol. (Of course, there is also a price exacted in terms of memory space required.) Listing 4 gives an example of a program that spends relatively more time just examining characters than do typical programs. The cost per character is 0.08 milliseconds ((35-27)/100,000 extra characters) for the standard BASIC and 0.11 ms for the extended form of BASIC. That is about 0.03 ms or 30 microseconds more in the extended BASIC, in agreement with the above mentioned 36 microseconds.

I assembled this code using the "Ohio Scientific 6500 Assembler/Editor" and put it at $C000 where I have 4K of memory (2102's on my old style 500 board). However, I suggest you put it at $0222 in page 2 because this space is otherwise unused by our non-disk machines. You can tape it in the "Auto-load" mode using a program such as that given by Bruce Hoyt in MICRO 11:17. Then the drill for use is this: Cold start BASIC and break to the monitor. Autoload the code of Listing 2 with the start address at $0222. The tape will finish loading, modify zero page and jump to warm start of BASIC. It's easy (if your tape loader doesn't put noise into $0000 as mine sometimes does).

I have a strong suspicion that I may have done some of these things the hard way. I would enjoy seeing some more elegant solutions to this "hooking". Even more practically, I would like to copy some useful extensions to BASIC from some future issues of MICRO. So how about it out there in 6502 land?

µ

```
 1 REM ***** TIMING TEST PROGRAM *****
 2 REM
 5 B=1
 10 FOR I=1 TO 10000
 20 A=B
 50 NEXT
 90 REM
 100 REM THE ABOVE PROGRAM IS THE "SHORT FORM"
 101 REM THE LONG FORM REPLACES 2 LINES WITH:
 102 REM 5 BC=1
 103 REM 20 ABABAB=BCBCBC
 104 REM THE LONG FORM HAS 10 MORE CHARACTERS TO PROCESS
 105 REM DURING EACH CYCLE OF THE LOOP
 106 REM
 110 REM THE TIMES (SECONDS) TO RUN THE PROGRAM
 111 REM USING THE STANDARD AND THE EXTENDED BASIC ARE
 112 REM
 114 REM            STANDARD   EXTENDED
 115 REM SHORT         27         29
 116 REM LONG          35         40
OK
```

Edward Carlson is a professor of Physics at Michigan State University. He has been doing job-related programming for Michigan State since 1961, and he is especially interested in Fortran and Machine languages.

At the moment he is using graphics to teach Physics. He is planning to write more articles for Micro in the near future.

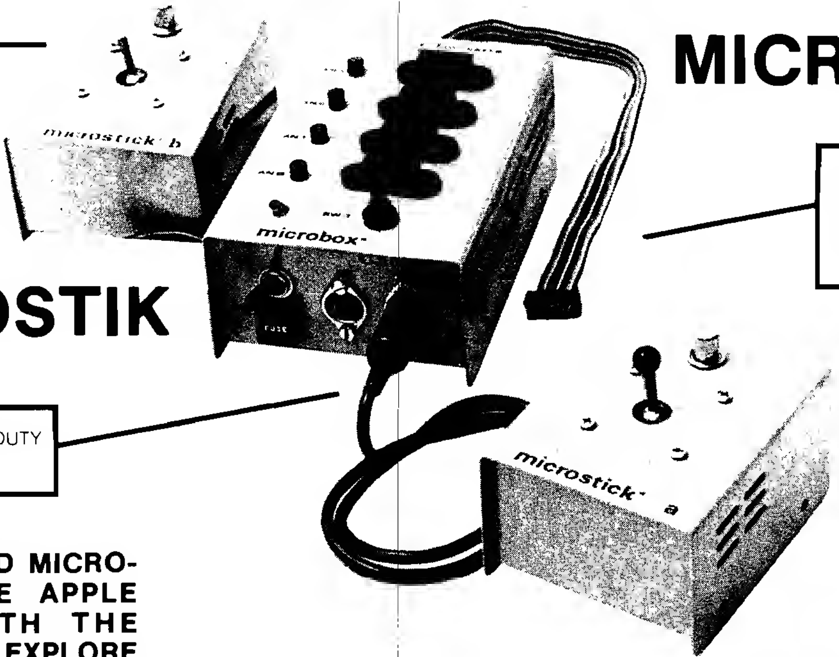
Now You Can Have

INPUT/OUTPUT

For Your Apple Computer

DUAL AXIS JOYSTICK AND PUSHBUTTON MODULE for Games, Graphics, and Experimental Program Input

MICROBOX



CONNECTS DIRECTLY to the Apple Game Socket ACCEPTS 2 Dual Axis MICROSTIKS or 4 Paddles

MICROSTIK

ISOLATED SWITCHING of 4 AC loads or relays from a basic program. 4 LED status indicators Toggle Switch input (sw3)

EXTRA LONG, HEAVY DUTY cables and connectors

MICROBOX AND MICROSTIK PROVIDE APPLE OWNERS WITH THE HARDWARE TO EXPLORE THE INPUT/OUTPUT CAPABILITIES OF THEIR COMPUTERS.

A SIMPLE COMMAND from the Apple keyboard or a Basic Program can switch an external device. Connect AC loads, such as lamps, motors, relays or solenoids directly through the MICROBOX's 4 AC OUTLETS. Loads can range from 0 to 220VAC and draw up to 200 Watts each. Solid State Switching ISOLATES the load from your Apple for complete safety. Four LEDs provide a visual on/off status of each load.

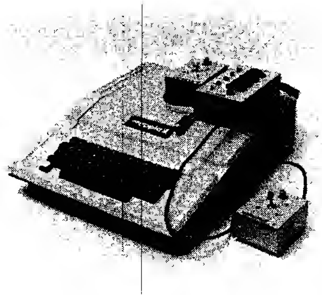
A Complete Instruction/Tutorial Manual is included with the MICROBOX.

CJM

INDUSTRIES

REAL-TIME INPUT

The MICROSTIK is a sturdy, two axis joystick. Metal Cable Connectors assure trouble free usage over time, and enable extension cables to be added easily. Use the MICROSTIK to add real-time input to your game, graphic or experimental programs. Each MICROSTIK contains a PUSHBUTTON for added input possibilities.



MICROBOX and MICROSTIK sit comfortably on, or aside the Apple Computer. They have been designed to match the Apple in color and design.

ORDER TODAY AND CONNECT YOUR APPLE TO THE OUTSIDE WORLD.

The MICROBOX and MICROSTIK can be purchased at most computer stores or can be ordered directly by mail or through our convenient 24 hour telephone service.

TELEPHONE:
(703) 620-2444

Order the MICROSET and receive the MICROBOX, 2 MICROSTIKS, the Manual and Cassette, and SAVE $40.

MICROBOX	**$139.95**
MICROSTIK (each)	**39.95**
Demo Cassette	**14.95**
MICROSET	**199.95**
12 ft Ext. cables	**9.95**
Relay Modules	**WRITE**
Solenoid Modules	**WRITE**

VA res. add 4% sales tax

MASTER CHARGE, VISA accepted
No C.O.D.'s

CJM Industries, Dept. MB
P.O. Box 2367
Reston, VA 22090

SYM-1 BASIC Pack Program

A Pack routine which permits comments to be removed from Basic to permit faster execution and to save space is presented for the SYM-1.

George Wells

One of the most important aspects of writing good BASIC programs is the quality of the documentation contained in the program. The excellent Hayden book , "BASIC with Style", by Nagine and Ledgard, sets forth a standard by which BASIC programs should be written. This standard makes liberal use of REMarks, spaces and blank and indented lines to highlight and bring out the logical structure of the program.(See Appendix B, Prettyprinting Standards, of "BASIC with Style.") Unfortunately, with the limited amount of RAM available on most micro-implemented BASICs, by the time you write such a program you may not have memory available to run it or you may have severely reduced the allowable sixes of matrices or strings.

Now, if you have a SYM-1 you can have the best of both worlds with this BASIC Pack program. If the assembly language program is stored on tape it can be loaded into page one and run, all without leaving BASIC. In just a couple seconds, it will delete all insignificant spaces, NULL lines and extra colons and reduce all variable names to two characters or less. The packed program will run exactly as its parent with one exception: GOTO's and GOSUB's are not allowed to point to Null lines since they are deleted. (A Null line is one consisting of only colons, spaces and/or a REMark.) However, the parent program must be error free or the packed program might end up with different errors. In most cases, you will notpack a program until it is completely debugged. If you do need to change a program after it is packed, you should reload the parent and change It. Then be sure to always save the unpacked version of the program on tape before packing or all your documentation work will be wasted.

The easiest way to use the Pack program is to make a copy of the object code on tape with an ID of $31. Verify the object code before and after saving to make sure the stack has not clobbered it. Then jump to BASIC and write and save your well-documented BASIC program. Next enter the BASIC direct command LOAD 1. Read the tape with the Pack program on it. After it has LOADED, enter ?USR(270,0). This will run the Pack program and return to BASIC with an OK. Now you can treat the packed program just like any other program.

Another way to use the Pack program is to assemble it yourself. If you have RAE-1 and 8K of RAM you can copy the Assembly Listing and assemble it. However, before you do enter the following command:

```
SET $200 $1CFC $1D00 $1F00
```

Incidentally, there is a minor error in RAE-1 as can be seen at address 137 of the Assembly Listing. Page 4-6 of the second printing of the RAE-1 Reference Manual states the current PC (=) is the first byte of the next instruction after a branch: instead it is one less than this.

Since the Resident Assembler/Editor uses parts of page one, the object code is stdred temporarily on page $1F. After you save the text on tape you must RESET the SYM-1 to get the stack pointer away from the end of the BASIC Pack program area and then move the code to page one with the monitor command M 1 0E,1F0E-1FE2. Now follow the same procedure given above for using the object code.

If you have EPROM in your system, you can assemble the program at some place in your EPROM or simply relocate the code to another page by changing all of the "01" bytes to the new page number. Of course, you will have to call the program by its correct address when you get ready to USR it.

To get an idea of what the BASIC pack program does a sample BASIC "program" is listed before and after packing. You should also be familiar with Appendix C, Space Hints, of the BASIC Reference Manual. One additional hint which is not mentioned is to use integer matrices instead of floating point matrices wherever possible. This saves 3 bytes per element: that's 363 bytes for DIM A %(10,10). There are other ways to save space: how about renumbering lines startng at zero and increasing by one or how about a program that determines which lines are not pointed to by any statements and packing consecutive such lines into long lines of up to 255 bytes each? These ideas are significantly more difficult to implement than those in this Pack program. For myself, I hope to have more memory before I need such a sophisticated program, but maybe someone else may take up the challenge to write one. μ

George Wells has been interested in computing since his high school days. He presently works at the Jet Propulsion Lab in Pasadena. Here he is employed in the instrumentation department where he makes custom designs for 6502-based systems microprocessors.

His wife, Marilyn, a registered nurse, does not share George's enthusiasm for computing, although their son, Bradley, age 5, loves to push the buttons. We understand that Bradley knows where the 'return' is located.

```
>ASSEMBLY LISTING

                    OUT.POINT   .DE $7D       POINTER TO OUTPUT MODIFIED TEXT
                    GET.RAM     .DE $CC       GET NEXT BYTE FROM RAM
                    BASIC.WARM  .DE $C27E     BASIC WARM START ENTRY
                    FIX.LIN.PN  .DE $C32C     FIX PROGRAM LINE POINTERS
                    RST.BAS.PN  .DE $C46D     RESET BASIC POINTERS (EOP+1 IN $7D)
                    IN.RAM.PNT  .DE $C49F     INITIALIZE GET.RAM POINTER
                    TEST.ALPHA  .DE $CEE9     TEST IF ALPHA (SET CARRY IF ALPHA)
                    REM         .DE $8E       REMARK TOKEN VALUE
                    DATA        .DE $83       DATA TOKEN VALUE

                                .BA $10E
                                .MC $1F0E
                                .OS

010E- 20 9F C4  BASIC.PACK JSR IN.RAM.PNT       INITIALIZE GET.RAM POINTER
0111- 85 7E                STA ◆OUT.POINT+1     COPY TO OUTPUT POINTER
0113- A5 D3                LDA ◆GET.RAM+7
0115- 85 7D                STA ◆OUT.POINT

0117- A2 FD     RST.STACK  LDX #$FD             RESET STACK POINTER
0119- 9A                   TXS
011A- A0 00                LDY #0               ALLOW GET.RAM TO GET SPACES
011C- 84 DC                STY ◆GET.RAM+16

011E- 20 CC 00  COPY.4BYTE JSR GET.RAM          COPY FOUR BYTES (POINTER
0121- C8                   INY                   PLUS LINE NUMBER)
0122- 91 7D                STA (OUT.POINT),Y
0124- C0 04                CPY #4
0126- 90 F6                BCC COPY.4BYTE       CARRY SET NEEDED LATER

0128- A9 EF                LDA #$EF             RESTORE GET.RAM TO IGNORE
012A- 85 DC                STA ◆GET.RAM+16       SPACES

012C- A0 02                LDY #2               CHECK FOR END OF PROGRAM
012E- B1 7D                LDA (OUT.POINT),Y
0130- D0 12                BNE NEW.LINE         BRANCH IF NOT EOP
0132- 98                   TYA                  A=Y=2 AND CARRY IS SET
0133- 65 7D                ADC ◆OUT.POINT       ADD 3 TO OUTPUT POINTER
0135- 85 7D                STA ◆OUT.POINT
0137- 90 02                BCC *+3
0139- E6 7E                INC ◆OUT.POINT+1     PROPAGATE CARRY
013B- 20 6D C4             JSR RST.BAS.PN       RESET BASIC POINTERS
013E- 20 2C C3             JSR FIX.LIN.PN       FIX PROGRAM LINE POINTERS
0141- 4C 7E C2             JMP BASIC.WARM       ENTER BASIC

0144- A0 04     NEW.LINE   LDY #4               SET Y TO FIRST BYTE IN LINE
0146- 20 BE 01  NEXT.BYTE  JSR GET.NEXTCQ       GET NEXT BYTE (COPY QUOTES)
0149- C8        CONT.BYTE  INY                  OUTPUT NEXT BYTE
014A- 91 7D                STA (OUT.POINT),Y
014C- B0 0D                BCS COLON.TEST       BRANCH IF NOT NUMERAL

014E- 20 B7 01  NUMERAL    JSR GET.OUTPUT       GET AND OUTPUT ANOTHER BYTE
0151- 90 FB                BCC NUMERAL          CONT TO LOOP IF NUMERAL
0153- C9 2E                CMP #'.              TEST FOR DECIMAL AND
0155- F0 F7                BEQ NUMERAL           CONT LOOP IF SO
0157- C9 45                CMP #'E              TEST FOR EXPONENT PART OF
0159- F0 EB                BEQ NEXT.BYTE         NUMERAL AND BRANCH IF SO

015B- C9 3A     COLON.TEST CMP #':              TEST FOR COLON AND
015D- D0 0A                BNE REM.TEST          BRANCH IF NOT
015F- 88                   DEY                  TEST FOR FIRST CHAR COLON
0160- C0 04                CPY #4
0162- F0 E2                BEQ NEXT.BYTE         AND IGNORE IF SO

0164- D1 7D                CMP (OUT.POINT),Y    TEST FOR COLON PAIR AND
0166- F0 DE                BEQ NEXT.BYTE         IGNORE IF SO
0168- C8                   INY                  OTHERWISE, ACCEPT COLON

0169- C9 8E     REM.TEST   CMP #REM             CHECK IF REM TOKEN
016B- D0 2A                BNE DATA.TEST         AND BRANCH IF NOT
016D- 88                   DEY                  IGNORE REM AND
016E- 88                   DEY                   PREVIOUS CHAR (COLON)
016F- 20 D2 01  FIND.EOL   JSR GET.NONBLK       INPUT TO END OF LINE
0172- 4C 6F 01             JMP FIND.EOL         DO NOT OUTPUT

0175- C8        OUTPUT.EOL INY                  OUTPUT END OF LINE
0176- 91 7D                STA (OUT.POINT),Y

0178- C0 06                CPY #6               TEST FOR NULL LINE AND
017A- 90 9B                BCC RST.STACK         START NEW LINE IF SO

017C- 98                   TYA                  CALCULATE OUTPUT POINTER
017D- 18                   CLC                   FOR NEXT LINE
017E- 65 7D                ADC ◆OUT.POINT
0180- 85 7D                STA ◆OUT.POINT
0182- 90 93                BCC RST.STACK        START NEW LINE
0184- E6 7E                INC ◆OUT.POINT+1     PROPAGATE CARRY
0186- B0 8F                BCS RST.STACK        (ALWAYS)

0188- 46 DA     DATA.SPACE LSR ◆GET.RAM+14      ALLOW GET.RAM TO GET SPACES
018A- 20 B7 01  DATA.CONT  JSR GET.OUTPUT       GET FIRST NONSPACE & OUTPUT
018D- C9 2C                CMP #',              TEST FOR END OF STRING
018F- F0 F9                BEQ DATA.CONT         AND BRANCH IF SO
0191- C9 3A                CMP #':              TEST FOR END OF STATEMENT
0193- F0 B1                BEQ NEXT.BYTE         AND BRANCH IF SO
0195- D0 E1                BNE DATA.SPACE       ELSE, ALLOW EMBEDDED SPACES

0197- C9 83     DATA.TEST  CMP #DATA            CHECK IF DATA TOKEN
0199- F0 EF                BEQ DATA.CONT         BRANCH IF SO

019B- 20 E9 CE  ALPHA.TEST JSR TEST.ALPHA       TEST IF CHAR IS ALPHA AND
019E- 90 A6                BCC NEXT.BYTE        BRANCH IF NOT VARIABLE NAME
01A0- 20 B7 01             JSR GET.OUTPUT       GET AND OUTPUT ANOTHER BYTE
01A3- 90 05                BCC FIND.EON         BRANCH IF NUMERAL
01A5- 20 E9 CE             JSR TEST.ALPHA       TEST IF ALPHA
01A8- 90 B1                BCC COLON.TEST       BRANCH IF EON-SINGLE ALPHA

01AA- 20 D2 01  FIND.EON   JSR GET.NONBLK       FIND END OF VARIABLE NAME
01AD- 90 FB                BCC FIND.EON         BRANCH IF NUMERAL
01AF- 20 E9 CE             JSR TEST.ALPHA
01B2- B0 F6                BCS FIND.EON         BRANCH IF ALPHA
01B4- 38                   SEC                  (TO TAKE NEXT TWO BRANCHES)
01B5- B0 92                BCS CONT.BYTE        END OF VARIABLE NAME

01B7- 20 BE 01  GET.OUTPUT JSR GET.NEXTCQ       GET NEXT BYTE (COPY QUOTES)
01BA- C8                   INY                  AND OUTPUT IT
01BB- 91 7D                STA (OUT.POINT),Y
01BD- 60                   RTS
                                               GET NON-BLANK CHAR FROM RAM
01BE- 20 D2 01  GET.NEXTCQ JSR GET.NONBLK      BRANCH (RTS) IF NOT QUOTE
01C1- D0 1F                BNE RTS             OUTPUT STRING INSIDE QUOTES
01C3- C8        OUT.QUOTES INY
01C4- 91 7D                STA (OUT.POINT),Y   GET NEXT CHAR-ALLOW SPACES
01C6- 20 D0 01             JSR GET.CHAR        BRANCH IF NOT END-QUOTE
01C9- D0 F8                BNE OUT.QUOTES
                                               OUTPUT END-QUOTE
01CB- C8                   INY
01CC- 91 7D                STA (OUT.POINT),Y   (ALWAYS) CONT TO GET CHAR
01CE- D0 02                BNE GET.NONBLK
```

```
01D0- 46 DA      GET NEXT CHAR FROM RAM          GET.CHAR    LSR *GET.RAM+14
                  (ALLOW SPACES)                 ;

01D2- 20 CC 00   GET NEXT CHAR FROM RAM          GET.NONBLK JSR GET.RAM
                  (SKIP SPACES)                  ;
01D5- 48                                                     PHA
01D6- A9 20      DISALLOW SPACES AGAIN                       LDA #'
01D8- 85 DA                                                  STA *GET.RAM+14
01DA- 68                                                     PLA
01DB- F0 98      BRANCH IF END OF LINE                       BEQ OUTPUT.EOL
01DD- 90 03      BRANCH IF NUMERAL                           BCC RTS
01DF- C9 22      SET EQUAL (Z) FLAG IF QUOTE                 CMP #'"
01E1- 38         SET NOT-NUMERAL FLAG                        SEC
01E2- 60                                         RTS         RTS
                                                             .EN

         LABEL FILE:  [ / = EXTERNAL ]


         /OUT.POINT=007D          /GET.RAM=00CC           /BASIC.WARM=C27E
         /FIX.LIN.PN=C32C         /RST.BAS.PN=C46D        /IN.RAM.PNT=C49F
         /TEST.ALPHA=CEE9         /REM=008E               /DATA=0083
         BASIC.PACK=010E          RST.STACK=0117          COPY.4BYTE=011E
         NEW.LINE=0144            NEXT.BYTE=0146          CONT.BYTE=0149
         NUMERAL=014E             COLON.TEST=015B         REM.TEST=0169
         FIND.EOL=016F            OUTPUT.EOL=0175         DATA.SPACE=0188
         DATA.CONT=018A           DATA.TEST=0197          ALPHA.TEST=019B
         FIND.EON=01AA            GET.OUTPUT=01B7         GET.NEXTCQ=01BE
         OUT.QUOTES=01C3          GET.CHAR=01D0           GET.NONBLK=01D2
         RTS=01E2
         //0000,01E3,1FE3
         >
```

OBJECT LISTING

```
.V 10E-1E2
010E 20 9F C4 85 7E A5 D3 85,83
0116 7D A2 FD 9A A0 00 84 DC,39
011E 20 CC 00 C8 91 7D C0 04,BF
0126 90 F6 A9 EF 85 DC A0 02,E0
012E B1 7D D0 12 98 65 7D 85,EF
0136 7D 90 02 E6 7E 20 6D C4,B3
013E 20 2C C3 4C 7E C2 A0 04,F2
0146 20 BE 01 C8 91 7D B0 0D,64
014E 20 B7 01 90 FB C9 2E F0,AE
0156 F7 C9 45 F0 EB C9 3A D0,61
015E 0A 88 C0 04 F0 E2 D1 7D,D7
0166 F0 DE C8 C9 8E D0 2A 88,46
016E 88 20 D2 01 4C 6F 01 C8,45
0176 91 7D C0 06 90 9B 98 18,F4
017E 65 7D 85 7D 90 93 E6 7E,5F
0186 B0 8F 46 DA 20 B7 01 C9,5F
018E 2C F0 F9 C9 3A F0 B1 D0,E8
0196 F1 C9 83 F0 EF 20 E9 CE,DB
019E 90 A6 20 B7 01 90 05 20,9E
01A6 E9 CE 90 B1 20 D2 01 90,19
01AE FB 20 E9 CE B0 F6 38 B0,79
01B6 92 20 BE 01 C8 91 7D 60,20
01BE 20 D2 01 D0 1F C8 91 7D,D8
01C6 20 D0 01 D0 F8 C8 91 7D,67
01CE D0 02 46 DA 20 CC 00 48,8D
01D6 A9 20 85 DA 68 F0 98 90,35
01DE 03 C9 22 38 60,BB
 70BB
.S2 31,10E-1E2
```

```
LIST

 100 REM      ** THIS IS A SAMPLE PROGRAM TO
 110 REM .    ** ILLUSTRATE HOW 'PACK' WORKS.
 120 :
 130 REM      PACK REMOVES UNNEEDED SPACES AND
 140 REM      COLONS AND SHORTENS UP VARIABLE
 150 REM      NAMES.
 160 :      FOR I = 1 TO 10
 170 :         FIRST(I) = MIDDLE(I) + LAST(I)
 180 :      NEXT I:     :    :  REM EXTRA COLONS
 190 REM      PACK DELETES NULL LINES
 200 REM      NULL LINES ARE THOSE THAT HAVE
 210 REM      NO EXECUTABLE STATEMENTS IN THEM.
 220 :           REM SUCH AS THIS ONE.
 240 REM      PACK WILL ALLOW SPACES BETWEEN
 250 REM      PAIRS OF QUOTES.
 260 PRINT "END OF LOOP"
 270 REM      PACK WILL ALLOW SIGNIFICANT
 280 REM      SPACES IN DATA STATEMENTS.
 290 DATA ONE,     TWO,     THREE,   FOUR
 300 DATA FIVE,    SIX,     SEVEN,   EIGHT
 310 DATA        THIS SENTENCE IS ONE STRING.
 320 DATA        "      THIS ONE HAS LEADING SPACES.
 330 REM      PACK WILL NOT CONFUSE THE 'E' IN
 340 REM      A CONSTANT WITH A VARIABLE NAME:
 350 A = 6.02 E23
 360 REM      COMPARE THE ABOVE WITH THE BELOW:
 370 A = 6.02*E23
 380 REM      *****WARNING*****
 390 REM      MAKE SURE ALL LINE REFERENCES ARE
 400 REM      TO LINES WITH EXECUTABLE STATEMENTS
 410 REM      EXAMPLE: THE FOLLOWING LINE WILL
 420 REM      PRODUCE AN ERROR.
 430 GOTO 420
OK
LOAD 1
LOADED
OK
?USR(270,0)

OK
LIST

 160 FORI=1TO10
 170 FI(I)=MI(I)+LA(I)
 180 NEXTI
 260 PRINT"END OF LOOP"
 290 DATAONE,TWO,THREE,FOUR
 300 DATAFIVE,SIX,SEVEN,EIGHT
 310 DATATHIS SENTENCE IS ONE STRING.
 320 DATA"     THIS ONE HAS LEADING SPACES."
 350 A=6.02E23
 370 A=6.02*E2
 430 GOTO420
OK
```

Southeastern Software

DATA CAPTURE 3.0

Is DATA CAPTURE just another smart terminal program for your Apple II™ or Apple II Plus™?

NO. It is a GENIUS TERMINAL PROGRAM and is designed to be used with the Micromodem II™.

Tired of watching data and programs scroll off the screen forever? Then DATA CAPTURE is the program for you.

- ANYTHING that appears on the screen of your Apple II can be captured. Any program or data.
- You can then save what you have captured to disk, dump it to your printer or even do simple editing with DATA CAPTURE.
- You can use DATA CAPTURE to compose text offline for later transmission to another computer. Think of the timeshare charges this will save you.
- Use DATA CAPTURE with the Dan Paymar Lower Case Adapter and you can enter lower case from the keyboard for transmission to another computer or capture both upper and lower case.
- A program is also included to convert your programs to text files for transmission using DATA CAPTURE.
- You receive two versions of the program. One is for upper case only and one for both upper and lower case use with the above adapter.

DATA CAPTURE will save you money if you are using a timesharing system because you can compose messages offline for later transmission. You can also quickly capture data for later reading, printing or editing. Requires DISK II™, APPLESOFT II™.

Price $29.95

If your local dealer does not have DATA CAPTURE then order directly. We ship DATA CAPTURE within 3 working days of receipt of order and welcome your personal check. We also accept Visa and Master Charge. Add $49.95 if you would also like to order the Dan Paymar Lower Case Adapter at the same time.

Ask for a catalog of our software.

* Apple II, Apple II Plus, Disk II and APPLESOFT II are trademarks of Apple Computer Company.

* Micromodem II is a trademark of D.C. Hayes Associates, Inc.

SOUTHEASTERN SOFTWARE
7270 Culpepper Drive
New Orleans, LA 70126

504/246-8438   504/246-7937

Share Your AIM Programs

Two features of the AIM, its 20 character printer and its built-in disassembler, are both a blessing and a bother. Together they make it very easy to dump uncommented listings. This program provides a method for combining the dissassembler output with comments.

Jody Nelis

I've always had a problem sharing machine language programs while working with a small system. To publish a program you need a hard copy listing which can be photo graphicaly reproduced. Even though my programs are fully documented, they are all in a hand written form and not submissable for publication.

Hand typed program listings are notorious for small significant errors. For that reason, most publishers require a program listing produced by the system it is running on. Even if I could get a hand typed listing accepted, it would probably take me longer to prepare it for publication than it took to write it. Fast, accurate typing is just not one one of my finer points.

The AIM is one up on many of the other single board systems in that it allows an error free mini-disassemble format and it has an on board printer to save the listings. It, however, lacks the ability to provide labels and comments with the listing. I feel that every published program should be fully documented to make it understandable. Full documentation also makes it useful to readers with other systems.

Many publishers agree with my feelings and don't publish hex dumps or AIM mini-dissassemble listings. For lack of any easy way of preparing a program listing, a lot of good programming remains buried in my fines and your files. Much of it is probably labeled and commented but, like mine it is in an unsubmissable form.

The program presented here will eliminate this frustration. Using this program, you can prepare a fully labeled and commented hard copy program listing with a minimum of effort. It will merge your working program in RAM with your labels and comments in the AIM text editor into a pseudo-standard format and output a fully documented listing to an external printer. You will have no excuse for keeping your programs in the closet!

Program Overview

To run this program you should have your AIM populated to 4K of RAM. However, it can be done with as little as 2K. You will need access to an external printer with a minimum of 60 characters per line. If you don't have such a printer, you could save this program, your program, and your program comments on cassette using your AIM, and take them to a friend's house to be printed on his AIM and printer combination. A publisher with an AIM and printer combination could also prepare a publishable listing from your cassette and instructions. The on board AIM can be used for editing your listing prior to making a final tape.

This program, which I shall call MERGE for short, occupies the last two pages of the onboard 4K RAM block. It is easily relocatable to suit your system by changing only the JSR and JMP arguments. It should reside in the top two pages of your system RAM.

When you have your particular program (which I will call the subject program) debugged and running the way it should, you are ready to prepare a documented listing. The normal AIM text editor commands are used to enter, edit and finalize the labels and comments for your subject program. This includes the use of the salvation of all poor typists the DELETE key! A coded entry is used to pack as much as possible into a limited capacity system.

MERGE will acomodate long label and comment listings by allowing multiple listings of editor data to cassette. Later, these blocks of data may be read back into the editor one at a time for use by MERGE.

MERGE is written for a continuous roll feed printer. It provides automatic paging with top and bottom margins for standard eleven inch sheets. Your program title and a sequencial page number is printed at the top of each sheet except the first.

Once you have MERGE and your subject program loaded and your labels and comments finalized and loaded in the text editor, MERGE will print out the documented program listing. It will be 100% correct as far as your subject program code is concerned and 99 to 100% correct as far as the labels and comments are concerned. The possible 1% error allowed is to cover a human-type operator's failings in proof reading the final listing.

Why This Program Was Written

Before undertaking the fun(?) of preparing this program, I took a quick look around to see if the answer already existed. If it does, it escaped me.

I looked into the AIM 4K Assembler ROM since there is a socket in my AIM all ready for it. From what I could gather, it is a dandy assembler but it's final listing is in two parts. While a source code listing and an object code listing could be worked together, it was not the solution I was after.

While I didn't scour the earth looking, I didn't run across anyone else making a ROM or EPROM which would plug into the assembler socket and do the job I wanted.And I didn't put much effort into examining any assembler-disassembler packages which would load into RAM since I don't have enough RAM to work with.

Trying to type the program listing into the text editor the way I wanted it to appear would just be a sneaky way of manually typing it. Not only would I make mistakes, but I would also pull my hair out trying to tab the columns while using the 20 character display. There also remained the problem of limited RAM. Spaces have a way of eating up editor RAM.

After throwing out all of these alternates out, it came down to the same old story; if you want it and can't find it, write it yourself!

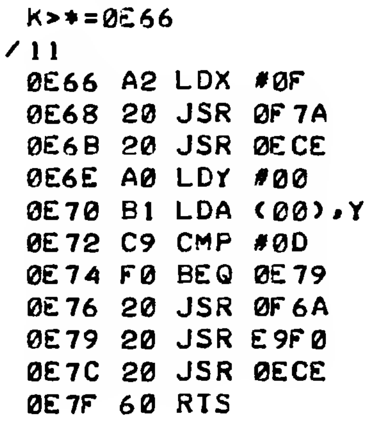
Program Description

MERGE, as presented here, represents an effort to create a good structured, top down program. I utilized subroutine calls freely to allow easy extraction if any routines should be of value in future programming efforts. I tried to use a lot of relative branching to make the program relocatable with a minimum of argument changes.

All of my subject programs seem to start at 0200 and grow upward as required. There is, ofcourse, a variable upper limit. Since my subject programs start at the bottom of the useful RAM, I put MERGE in the top of my RAM. It occupies 509 bytes from 0E00 to 0FFC.

This leaves a block of continuous RAM available between the top of the subject program and ODFF. This block is used by the AIM text editor to prepare the labels and comments for the subject program. To make maximum use of this limited text editor capacity, the text is condensed to eliminate spaces and semicolons wherever possible.

Figure 1 is the subroutine TITLE listed by the AIM monitor "K" command. Figure 2 is a text editor listing of the labels and comments I have prepared to accompany this subroutine. As you can see, the data in figure 2 is quite condensed to conserve any wasted space.



```
  K>*=0E66
/11
  0E66  A2  LDX  #0F
  0E68  20  JSR  0F7A
  0E6B  20  JSR  0ECE
  0E6E  A0  LDY  #00
  0E70  B1  LDA  (00),Y
  0E72  C9  CMP  #0D
  0E74  F0  BEQ  0E79
  0E76  20  JSR  0F6A
  0E79  20  JSR  E9F0
  0E7C  20  JSR  0ECE
  0E7F  60  RTS
```

***Figure 1:* Aim Mini-Disassemble Format Listing of 'Title' Subroutine**

Figure 3 is the marriage of the two listings performed by MERGE. The condensed label and comment data has been separated and tabbed into the correct columns. The comments have been co-ordinated with the disassembly listing so that they fall on the correct lines.

The rules for inputting the label and comment data into the AIM text editor are spelled out in figure 4. It may appear complex and confusing at first, but, actually it is really quite easy to get the hang of it. If any errors are made in inputing the data or line codes, the printout will soon show it. Corrections are easily made using the standard AIM text editor commands.

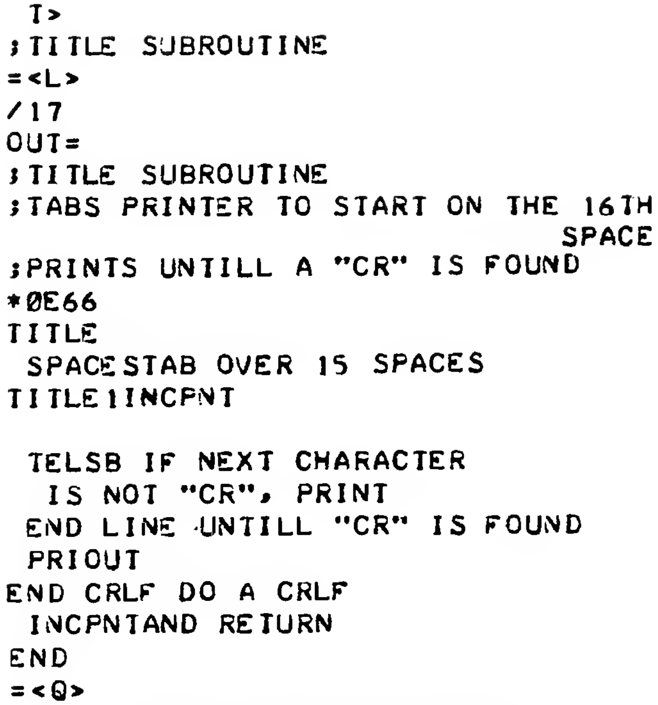
The complete program is listed in figure 5. MERGE has been used to produce it's own listing. Every available feature has been used and is illustrated in the listing.

The beauty of a fully labeled and commented program listing is that it pretty much tells it's own story. I need only comment on a fewhighlights of the various features here in the text.

The initialization portion of the listing sets as many registers as it can to their start values. Since the start address for the text editor varies with the size of the subject program, MERGE requests this information from the operator each time it is entered or re-entered with the prompt:"FROM = ". When a four digit address is entered from the keyboard followed by RETURN, the text editor start address is stored in a register and MERGE starts outputing data.

The brains for MERGE reside in the MAIN CONTROL LOOP which analyzes the first character of each data line in the text editor. It decodes the first character and calls upon the proper subroutine to format and print that line until the CR signifying the end of the line is encountered. The program then returns to the MAIN CONTROL LOOP to handle succeeding lines similarly. When the text editor end of data marker (00) is found, MERGE exits to the AIM monitor. If more data exists to finish the program listing, the text editor is reloaded from tape and MERGE is re-entered in a way not to disturb any of the paging registers. MERGE commences to print the new data until it once again finds an end of data marker.

The CHANGE DISASSEMBLE ADDRESS SUBROUTINE provides the co-ordination between the data in the text editor and the subject program. When as astrix followed by a four digit hex address is encountered in a line in the editor, the pointers to the next instruction to be disassembled are changed to that address. Otherwise, the next consecutive program step is disassembled.



```
 T>
;TITLE SUBROUTINE
=<L>
/17
OUT=
;TITLE SUBROUTINE
;TABS PRINTER TO START ON THE 16TH
                             SPACE
;PRINTS UNTILL A "CR" IS FOUND
*ØE66
TITLE
 SPACESTAB OVER 15 SPACES
TITLE1INCPNT

 TELSB IF NEXT CHARACTER
  IS NOT "CR", PRINT
 END LINE UNTILL "CR" IS FOUND
 PRIOUT
END CRLF DO A CRLF
 INCPNTAND RETURN
END
=<Q>
<
```

Figure 2: **Aim text Editor Listing of Labels and Comments Prepared for 'Title' Subroutine**

Automatic paging with titles and page numbers is provided by subroutine PAGE. The routine is written to provide top and bottom margins for a standard eleven inch long page with six lines per inch. If your printer uses any other line spacing, you will have to adjust the argument for the instruction at 0E9B which determines if it is time to start a new page or not. Make a corresponding change to the argument for the instruction at 0E00 which initializes PAGE for six line feeds prior to starting the first page. It should be one less than the byte at 0E9B.

As fully explained in figure 4, re-entering MERGE with multiple text editor loads of labels and comments may be required when doing long programs with short memory. A cut and splice must be performed on the printout to eliminate the extraneous lines of data printed when re-entering. When the copy has been spliced, the paging continuity is restored. If for some reason, you want a continuous listing without the paging feature, NOP the JSR PAGE at 0E19.

I've found that the secret to getting the most out of a small system like the AIM is by being familiar with the monitor ROM. I use as many subroutine calls to the monitor as I can to keep my programs short. While monitor subroutines are easy to get into, they don't always exit just when you want them to. When that happens, lift out the portions of the routine that you can use and rewrite it into your program. I've used portions of the monitor code, massaged to suit my needs, in my subroutines MERGE and ASCII TABLE PRINTOUT. I'm actually utilizing portions of the monitor"K" and "M" commands to achieve my listing. I had to handle their entry and exit differently and also control the pointers differently to get my desired results.

A very useful subroutine is MESSAGE. I use this routine regularly for prompts and comments in my interactive programs. It was derived from an almost identical monitor routine. The monitor routine, however, is locked into a message table in ROM and is of no use for direct subroutine calls. Put it in RAM as I have done. Set up your message table somewhere else in RAM and call with X set to the start of the message you want. MESSAGE will sequencially out put the bytes in the table to the display/printer until it encounters a stop byte. The stop byte will have it's MSB set to a one. For example, an ASCII space is 20 hex. To make this a stop byte, change it to a AO hex in the table. A message table can be up to 256 characters long using indexed addressing or longer using indirect indexed addressing. Type ASCII messages into the memory table using the AIM text editor normally. Then go back and locate the end of the individual messages using the "M" command. Change the last byte of each message to a stop byte with "/" command.

No special tricks are used in MERGE; it fit into the two pages I had allocated for it without having to get fancy. By keeping all of the subroutines intact and separate, and maybe even a little redundent in places, it should be easier to follow and understand.

Operating Instructions

Type in Merge from figure 5 and save 006 to 0015, 010C to 0111 and 0E00 to 0FFC on tape. The F2 key has been initialized for initial entry of MERGE. The F1 key has been initialized for the re-entry point when using multiple text editor data loads. Refer to figure 4 for complete operating instructions.

Subheading: Further Enhancements

Since MERGE relies on finding coded data in the text editor and merely processes it to the desired format, there is not much more you could do with it. You could adjust the tab locations to get wider titles or maybe let MERGE format the MONITOR EQUATES and REGISTERS USED sections. I chose to do them manually to avoid having too many confusing input coding designations. You could change the "CONTROL S" code to a "CONTROL B" code if that helps you remember a blank line easier.

If you have your tape recorder running under remote control, you could write a patch which would let MERGE reload the text editor with a new section of data when required. This would save you the chore of doing it manually each time. You would have to tape all of the data sections in the proper sequence. They would all have to be loaded into the editor at the same start address. Add another flag at the end of the last data section to let MERGE know that it was totally finished with the listing and may exit to the AIM monitor. With 110 baud printers and long programs, you could start MERGE up and walk away for hours while it prints. Maybe you could even visit your family for a while!

Another possibility might be a small program to control the data input into the editor. It would be handy to control the maximum comment length to avoid exceeding the width of your printer. This could reside in the same RAM location that MERGE resides in since only one at a time would be used.

Summary

With the availability of MERGE, every AIM-65 owner now has a memory efficient and easy to use means of listing his fully documented program. MERGE will be usefull to you even if you don't have an external printer. Your subject program, your labels, comments and MERGE can all be saved on cassette. Another AIM owner with a printer or a publisher with an AIM/printer can then produce the publishable listing from your cassette.

Sharing programs benefits us all. If we can eliminate duplicate efforts, we can concentrate on a new application. I'm looking forward to seeing your favorite program in print. While I probably won't be able to use all of them exactly as written, I am sure I'll learn a programming trick or two and be able to lift some of your subroutines out for my use. Get them documented and listed by MERGE and get them in the mail.

μ

Figure 3: **Formatted Listing of 'Title' Subroutine Produced by 'MERGE'.Date in Figure 1 and Figure 2 has been combined**

```
]>FROM=0200

                  ;TITLE SUBROUTINE
                  ;TABS PRINTER TO START ON THE 16TH SPACE
                  ;PRINTS UNTILL A "CR" IS FOUND
                  *=0E66
TITLE   0E66 A2 LDX #0F
        0E68 20 JSR 0F7A    SPACES   ;TAB OVER 15 SPACES
TITLE1  0E6B 20 JSR 0ECE    INCPNT
        0E6E A0 LDY #00
        0E70 B1 LDA (00),Y  TELSB    ;IF NEXT CHARACTER
        0E72 C9 CMP #0D              ;IS NOT "CR", PRINT
        0E74 F0 BEQ 0E79    END      ;LINE UNTILL "CR" IS FOUND
        0E76 20 JSR 0F6A    PRIOUT
END     0E79 20 JSR E9F0    CRLF     ;DO A CRLF
        0E7C 20 JSR 0ECE    INCPNT   ;AND RETURN
        0E7F 60 RTS
```

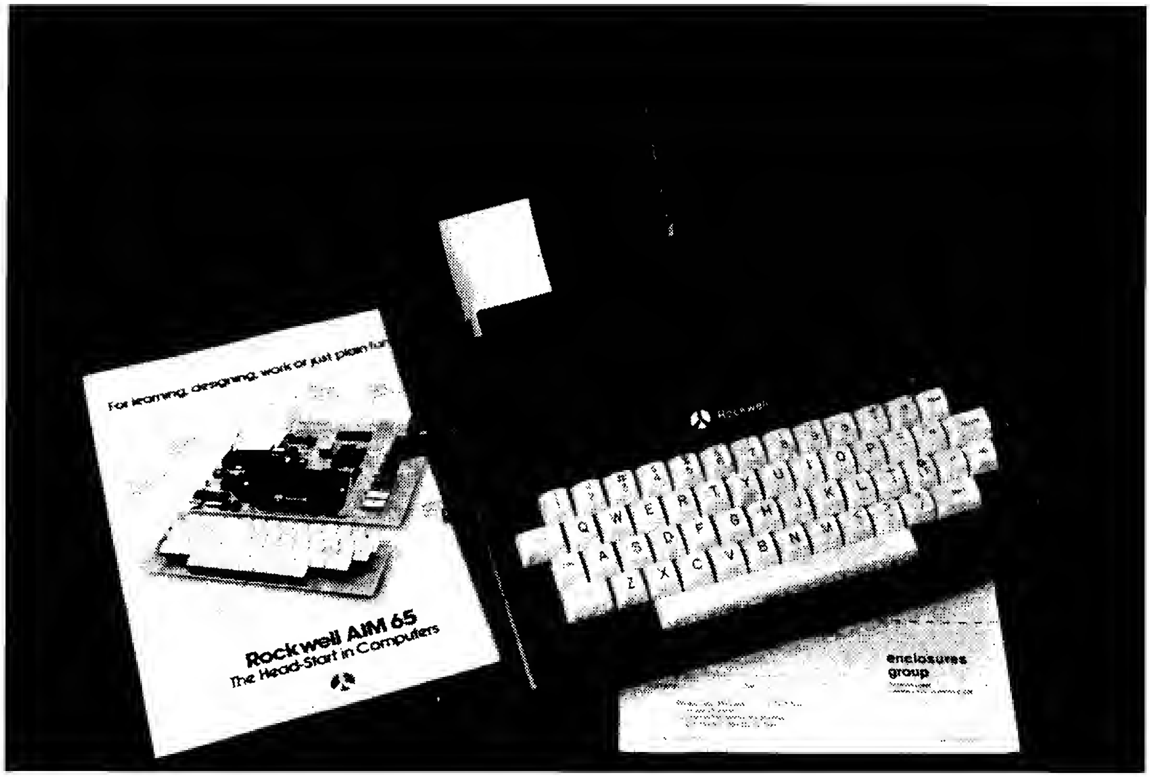
Figure 4

```
<

MERGE COMMENTS WITH DISASSEMBLY
*******************************

MERGES LABELS UP TO 6 CHARACTERS
AND COMMENTS UP TO 35 CHARACTERS
WITH AIM MINI DISASSEMBLE FORMAT
FOR PRINTING TO EXTERNAL PRINTER
USING 8 1/2" WIDE CONTINUOUS
FEED ROLL PAPER-PROVIDES PAGING

BY JODY NELIS, K3JZU
DECEMBER, 1979

REGISTERS USED
**************
0000 TELSB  ;TEXT EDITOR POINTER
0001 TEMSB  ;TEXT EDITOR POINTER
0002 LCOUNT ;LINE COUNTER
0003 PCOUNT ;PAGE COUNTER
0004 TITLSB ;TITLE POINTER
0005 TITMSB ;TITLE POINTER
00EA LENGTH ;BYTES IN ARGUMENT
A415 CURPOS ;CURSOR POSITION
A419 COUNT  ;DISASSEMBLE LINE COUNT
A41C ADDR   ;ADDRESS POINTER
A41D ADDR+1 ;ADDRESS POINTER
A425 SAVPC  ;PROGRAM COUNTER SAVE
A426 SVPC+1 ;PROGRAM COUNTER SAVE
```

```
                  MONITOR EQUATES
                  ***************
                  E1A1 COMIN  ;RETURN TO MONITOR
                  E610 NXT4   ;PRINT NEXT 4 BYTES
                  E615 NOW4   ;PRINT 4 BYTES
                  E7A3 FROM   ;INPUT 4 DIGIT ADR
                  E7D8 EQUAL  ;OUTPUT = SIGN
                  E83B BLANK2 ;OUTPUT 2 SPACES
                  E83E BLANK  ;OUTPUT 1 SPACE
                  E97A OUTPUT ;OUTPUT ASCII BYTE
                  EA46 NUMA   ;OUTPUT HEX BYTE
                  EA84 PACK   ;2 ASCII BYTES INTO 1 HEX BYTE
                  EB44 CLR    ;CLEAR DISPLAY POINTER
                  EF90 CRLF   ;OUTPUT A CRLF
                  F46C DISASM ;DISASSEMBLE 1 LINE

                  *=010C
                  ;INITIALIZE USER KEYS
F1      010C 4C JMP 0E5F    RE-ENT   ;RE-ENTER MAIN ROUTINE
F2      010F 4C JMP 0E00    ENTER    ;ENTER MAIN ROUTINE

                  *=0E00
                  ;INITIALIZE REGISTERS
INIT    0E00 A9 LDA #39              ;SET UP LINE COUNTER
        0E02 85 STA 02      LCOUNT   ;FOR INITIAL FEED
        0E04 A9 LDA #00
        0E06 85 STA 03      PCOUNT   ;SET PAGE COUNTER TO ZER(
RENTER  0E08 20 JSR E7A3    FROM
        0E0B AD LDA A41C    ADDR     ;GET THE TEXT EDITOR
```

Continued on page 29

PERFECT AIM


For learning, designing, work or just plain fun
Rockwell AIM 65
The Head-Start in Computers
Rockwell
enclosures group

ATTRACTIVE FUNCTIONAL PACKAGING FOR YOUR AIM-65 MICROCOMPUTER

- Professional Appearance
- Striking Grey and Black Color Combination
- Protects Vital Components

ENGINEERED SPECIFICALLY FOR THE ROCKWELL AIM-65

- All Switches Accessible
- Integral Reset Button Actuator
- Easy Paper Tape Replacement

EASILY ASSEMBLED

- Absolutely No Alteration of AIM-65 Required
- All Fasteners Provided
- Goes Together in Minutes

MADE OF HIGH IMPACT STRENGTH THERMOFORMED PLASTIC

- Kydex 100*
- Durable
- Molded-In Color
- Non-Conductive

AVAILABLE FROM STOCK

- Allow Three to Four Weeks for Processing and Delivery
- No COD's Please
- Dealer Inquiries Invited

TO ORDER: 1. Fill in this Coupon (Print or Type Please)
2. Attach Check or Money Order and Mail to:

NAME ____________________

STREET ____________________

CITY ____________________

STATE ____________________ ZIP __________

SAE 1-1 PLEASE SHIP PREPAID ________ SAE 1-1(s)
@ $43.50 each
California Residents Please Pay
$46.33 (Includes Sales Tax)

SAE 1-2 PLEASE SHIP PREPAID ________ SAE 1-2(s)
@ $46.50 each
California Residents Please Pay
$49.52 (Includes Sales Tax)

enclosures group

771 bush street
san francisco, california 94108

***TM Rohm & Hass** **Patent Applied For**

```
        0EDB 68 PLA
        0EDC 20 JSR E97A      OUTPUT   ;PRINT "*=" FOLLOWED
        0EDF 20 JSR E7D8      EQUAL    ;BY FOUR DIGIT
        0EE2 20 JSR 0ECE      INCPNT   ;ADDRESS
        0EE5 A0 LDY #00
ADDOUT  0EE7 B1 LDA (00),Y    TELSB
        0EE9 20 JSR E97A      OUTPUT
        0EEC C8 INY
        0EED C0 CPY #04
        0EEF D0 BNE 0EE7      ADDOUT
        0EF1 A0 LDY #00
        0EF3 20 JSR 0F06      CONVRT   ;CONVERT 4 DIGIT ASCII
        0EF6 8D STA A426      SVPC+1   ;ADR TO 2 BYTE HEX ADR
        0EF9 20 JSR 0F06      CONVRT   ;& PUT IT IN PROGRAM
        0EFC 8D STA A425      SAVPC    ;COUNTER REGISTER
        0EFF 20 JSR E9F0      CRLF
        0F02 20 JSR 0ECE      INCPNT
        0F05 60 RTS                    ;LINE FINISHED

                ;CONVERT SUBROUTINE
                ;CONVERTS 2 ASCII BYTES IN TEXT
                ;EDITOR TO ONE HEX BYTE
CONVRT  0F06 A2 LDX #02
CONVR1  0F08 B1 LDA (00),Y    TELSB    ;GET 2 ASCII CHARACTERS
        0F0A 20 JSR EA84      PACK     ;FROM TEXT EDITOR AND
        0F0D 20 JSR 0ECE      INCPNT   ;PACK THEM INTO ONE
        0F10 CA DEX                    ;HEX BYTE
        0F11 D0 BNE 0F08      CONVR1
        0F13 60 RTS                    ;RETURN WITH BYTE IN ACCUM

                ;ASCII TABLE PRINTOUT SUBROUTINE
                ;PRINTS HEX BYTES FOLLOWED BY ASCII CHARACTERS
                ;AND COMMENTS (IF 2ND CHAR IN LINE IS A "*", THE
                ;NEXT 4 CHARS ARE START ADR FOR THE TABLE)
ASCIIT  0F14 20 JSR 0ECE      INCPNT
        0F17 B1 LDA (00),Y    TELSB    ;IF 2ND CHAR IS NOT
        0F19 C9 CMP #2A                ;"*", DO NEXT 4 BYTES
        0F1B D0 BNE 0F37      CONTIN
        0F1D 20 JSR 0ECE      INCPNT   ;IF IT IS "*", SET
        0F20 20 JSR 0F06      CONVRT   ;UP START ADR
        0F23 8D STA A41D      ADDR+1   ;FOR TABLE
        0F26 20 JSR 0F06      CONVRT
        0F29 8D STA A41C      ADDR
        0F2C A2 LDX #09
        0F2E 20 JSR 0FE3      MESS     ;PRINT OUT "<M>="
        0F31 20 JSR E615      NOW4     ;AND FIRST FOUR BYTES
        0F34 38 SEC
        0F35 B0 BCS 0F3F      DECODE   ;FORCED JUMP TO DECODE
CONTIN  0F37 A2 LDX #09
        0F39 20 JSR 0FE3      MESS     ;PRINT OUT "<M>="
        0F3C 20 JSR E610      NXT4     ;AND NEXT FOUR BYTES
DECODE  0F3F 20 JSR E83E      BLANK    ;TAB ONE SPACE
        0F42 A2 LDX #04
        0F44 A0 LDY #00
DECOD1  0F46 20 JSR E83E      BLANK    ;PRINT OUT 4 ASCII
        0F49 B1 LDA (00),Y    TELSB    ;CHARACTERS
        0F4B 20 JSR E97A      OUTPUT   ;SEPARATED BY
        0F4E 20 JSR 0ECE      INCPNT   ;SPACES
        0F51 CA DEX
        0F52 D0 BNE 0F46      DECOD1
        0F54 20 JSR E83B      BLANK2   ;TAB TWO SPACES
        0F57 B1 LDA (00),Y    TELSB    ;IF NEXT CHARACTER
        0F59 C9 CMP #0D                ;IS NOT A "CR",
```

```
        0F5B F0 BEQ 0F63      LDONE    ;PRINT A ";"
        0F5D 20 JSR E9BA      SEMI     ;FOLLOWED BY COMMENTS
        0F60 20 JSR 0F6A      PRIOUT
LDONE   0F63 20 JSR E9F0      CRLF     ;DO A CRLF,
        0F66 20 JSR 0ECE      INCPNT   ;ADJUST POINTER &
        0F69 60 RTS                    ;RETURN

                ;PRINTOUT SUBROUTINE
                ;PRINTS OUT CHARS UNTILL A "CR" IS ENCOUNTERED
                ;(CALL WITH POINTER SET TO CHAR TO BE PRINTED)
PRIOUT  0F6A B1 LDA (00),Y    TELSB    ;GET CHARACTER
        0F6C C9 CMP #0D                ;IF IT IS A "CR", RETURN
        0F6E F0 BEQ 0F79      ENDLIN
        0F70 20 JSR E97A      OUTPUT   ;IF IT IS NOT "CR",
        0F73 20 JSR 0ECE      INCPNT   ;PRINT IT & GET
        0F76 38 SEC                    ;NEXT CHARACTER
        0F77 B0 BCS 0F6A      PRIOUT
ENDLIN  0F79 60 RTS                    ;RETURN

                ;SPACES SUBROUTINE
                ;PRINTS NUMBER OF SPACES IN X REGISTER WHEN CALLED
SPACES  0F7A 20 JSR E83E      BLANK    ;OUTPUT SPACES
        0F7D CA DEX                    ;UNTILL X=0
        0F7E D0 BNE 0F7A      SPACES
        0F80 60 RTS                    ;RETURN

                ;MERGE SUBROUTINE
                ;COMBINES LABELS & COMMENTS IN TEXT EDITOR
                ;WITH AIM MNEMONIC DISASSEMBLY OF OBJECT CODE
MERGE   0F81 A9 LDA #01                ;SET UP TO DISASSEMBLE 1 LINE
        0F83 8D STA A419      COUNT
        0F86 20 JSR EB44      CLR      ;CLEAR CURSOR POSITION REGISTER
        0F89 20 JSR 0FC6      GETLBL   ;PRINT PROGRAM LABEL OR SPACES
DISASS  0F8C 20 JSR F46C      DISASM   ;DISASSEMBLE ONE PROGRAM LINE
        0F8F AD LDA A425      SAVPC    ;IN AIM-65 FORMAT
        0F92 38 SEC
        0F93 65 ADC EA        LENGTH   ;INCREMENT PROGRAM COUNTER
        0F95 8D STA A425      SAVPC    ;REGISTERS TO POINT TO
        0F98 90 BCC 0F9D      CHECK    ;NEXT PROGRAM LINE
        0F9A EE INC A426      SVPC+1
CHECK   0F9D AD LDA A415      CURPOS   ;GET CURSOR POSITION
        0FA0 C9 CMP #1A
        0FA2 F0 BEQ 0FAB      ARGLBL   ;IF NOT AT 27TH POSITION
        0FA4 A9 LDA #20                ;IN LINE, TAB TO THERE
        0FA6 20 JSR E97A      OUTPUT
        0FA9 D0 BNE 0F9D      CHECK
ARGLBL  0FAB 20 JSR 0FF0      PATCH1   ;PRINT ARGUMENT LABEL OR SPACES
        0FAE 20 JSR E83B      BLANK
        0FB1 B1 LDA (00),Y    TELSB    ;GET NEXT CHARACTER
        0FB3 C9 CMP #0D
        0FB5 F0 BEQ 0FBF      DONE     ;END LINE IF IT IS A "CR"
        0FB7 A9 LDA #3B
        0FB9 20 JSR E97A      OUTPUT
        0FBC 20 JSR 0F6A      PRIOUT   ;OTHERWISE PRINT A ";"
DONE    0FBF 20 JSR E9F0      CRLF     ;FOLLOWED BY THE COMMENTS
        0FC2 20 JSR 0ECE      INCPNT   ;FOR THIS LINE & RETURN
        0FC5 60 RTS

                ;GET LABEL SUBROUTINE
                ;PRINTS A LABEL FROM DATA IN TEXT EDITOR
                ;DEFAULTS TO 6 SPACES IF A SPACE ENCOUNTERED
GETLBL  0FC6 A2 LDX #06                ;LABEL 6 CHARACTERS LONG
GECHAR  0FC8 B1 LDA (00),Y TELSB       ;GET CHARACTER
```

```
        0E0E 85 STA 00       TELSB    ;START ADR AND PUT
        0E10 85 STA 04       TITLSB   ;IT IN THE REGISTERS
        0E12 AD LDA A41D     ADDR+1
        0E15 85 STA 01       TEMSB
        0E17 85 STA 05       TITMSB

                 ;MAIN CONTROL LOOP
                 ;EXAMINES THE FIRST CHARACTER OF EACH LINE
                 ;AND DETERMINES PROPER PRINTING FORMAT
MAIN    0E19 20 JSR 0E97     PAGE     ;START NEW PAGE AS REQ'D
        0E1C A0 LDY #00
        0E1E B1 LDA (00),Y   TELSB    ;GET FIRST CHAR IN LINE
        0E20 C9 CMP #13
        0E22 D0 BNE 0E2A     ASTRIX   ;PRINT OUT BLANK LINE
        0E24 20 JSR 0E8D     BLANK    ;IF IT IS A CONTROL "S"
        0E27 4C JMP 0E19     MAIN
ASTRIX  0E2A C9 CMP #2A
        0E2C D0 BNE 0E34     CONTRA   ;INPUT DISASSEMBLE START
        0E2E 20 JSR 0ED5     ADDRIN   ;ADDRESS IF IT IS AN ASTRIX
        0E31 4C JMP 0E19     MAIN
CONTRA  0E34 C9 CMP #01
        0E36 D0 BNE 0E3E     SEMCOL   ;PRINT OUT ASCII TABLE LINE
        0E38 20 JSR 0F14     ASCIIT   ;IF IT IS A CONTROL "A"
        0E3B 4C JMP 0E19     MAIN
SEMCOL  0E3E C9 CMP #3B
        0E40 D0 BNE 0E48     CONTRT   ;TAB IN & PRINT COMMENT
        0E42 20 JSR 0E80     COMENT   ;LINE IF IT IS A ";"
        0E45 4C JMP 0E19     MAIN
CONTRT  0E48 C9 CMP #14
        0E4A D0 BNE 0E52     END      ;TAB IN & PRINT TITLE LINE
        0E4C 20 JSR 0E66     TITLE    ;IF IT IS A CONTROL "T"
        0E4F 4C JMP 0E19     MAIN
END     0E52 C9 CMP #00               ;RETURN TO MONITOR IF END
        0E54 F0 BEQ 0E5C     ENDDIS
        0E56 20 JSR 0F81     MERGE    ;DISASSEMBLE LINE AND
        0E59 4C JMP 0E19     MAIN     ;MERGE COMMENTS
ENDDIS  0E5C 4C JMP E1A1     COMIN    ;RETURN TO MONITOR

                 ;RE-ENTER MAIN CONTROL LOOP
                 ;USED WHEN MEMORY WILL NOT HOLD ALL
                 ;OF THE COMMENTS AND LABELS AT ONCE
                 ;(FIRST LINE OF ALL SECTIONS MUST BE THE TITLE)
RE-ENT  0E5F 20 JSR 0FF6     FATCH2   ;DECREMENT LINE COUNT SO TITLE
        0E62 EA NOP                   ;LINE WILL NOT COUNT IN PAGING
        0E63 4C JMP 0E08     RENTER

                 ;TITLE SUBROUTINE
                 ;TABS PRINTER TO START ON THE 16TH SPACE
                 ;PRINTS UNTILL A "CR" IS FOUND
TITLE   0E66 A2 LDX #0F
        0E68 20 JSR 0F7A     SPACES   ;TAB OVER 15 SPACES
TITLE1  0E6B 20 JSR 0ECE     INCPNT
        0E6E A0 LDY #00
        0E70 B1 LDA (00),Y   TELSB    ;IF NEXT CHARACTER
        0E72 C9 CMP #0D               ;IS NOT "CR", PRINT
        0E74 F0 BEQ 0E79     END      ;LINE UNTILL "CR" IS FOUND
        0E76 20 JSR 0F6A     PRTOUT
END     0E79 20 JSR E9F0     CRLF     ;DO A CRLF
```

```
        0E7C 20 JSR 0ECE     INCPNT   ;AND RETURN
        0E7F 60 RTS

                 ;COMMENT LINE SUBROUTINE
                 ;TABS PRINTER TO START ON 16TH SPACE WITH A ";"
                 ;PRINTS UNTILL A "CR" IS FOUND
COMENT  0E80 48 PHA                   ;SAVE SEMICOLON
        0E81 A2 LDX #0F
        0E83 20 JSR 0F7A     SPACES   ;TAB OVER 15 SPACES
        0E86 68 PLA
        0E87 20 JSR E97A     OUTPUT   ;PRINT SEMICOLON
        0E8A 38 SEC
        0E8B B0 BCS 0E6B     TITLE1   ;PRINT REMAINDER OF LINE

                 ;BLANK SUBROUTINE
                 ;INSERTS A BLANK LINE IN THE PRINTOUT
BLANK   0E8D 20 JSR E9F0     CRLF     ;DO A CRLF
        0E90 20 JSR 0ECE     INCPNT   ;INCREMENT POINTER
        0E93 20 JSR 0ECE     INCPNT
        0E96 60 RTS                   ;RETURN

                 ;PAGE SUBROUTINE
                 ;PROVIDES AUTOMATIC PAGING WITH
                 ;TITLES AND PAGE NUMBERS
PAGE    0E97 E6 INC 02       LCOUNT   ;INCREMENT LINE COUNTER
        0E99 A5 LDA 02       LCOUNT
        0E9B C9 CMP #3A
        0E9D D0 BNE 0ECD     NOTREQ   ;IF 58 LINES HAVE
        0E9F A2 LDX #06               ;BEEN PRINTED OUT,
LINES   0EA1 20 JSR E9F0     CRLF     ;INCREMENT THE COPY
        0EA4 CA DEX                   ;SIX LINES
        0EA5 D0 BNE 0EA1     LINES
        0EA7 86 STX 02       LCOUNT   ;ZERO LINE COUNTER
        0EA9 E6 INC 03       PCOUNT
        0EAB A5 LDA 03       PCOUNT   ;INCREMENT PAGE COUNTER
        0EAD C9 CMP #01
        0EAF F0 BEQ 0ECD     NOTREQ   ;IF PAGE 1, NO TITLE REQ'D
        0EB1 8A TXA
        0EB2 A8 TAY                   ;ZERO OFFSET
TITLE   0EB3 B1 LDA (04),Y   TITLSB
        0EB5 C9 CMP #0D               ;PRINT PROGRAM TITLE
        0EB7 F0 BEQ 0EBF     FINISH   ;AT TOP OF PAGE
        0EB9 20 JSR E97A     OUTPUT
        0EBC C8 INY
        0EBD D0 BNE 0EB3     TITLE
FINISH  0EBF 20 JSR 0FE3     MESS     ;PRINT PAGE NUMBER
        0EC2 A5 LDA 03       PCOUNT   ;AT TOP OF PAGE
        0EC4 20 JSR EA46     NUMA
        0EC7 20 JSR E9F0     CRLF     ;INCREMENT THE COPY
        0ECA 20 JSR E9F0     CRLF     ;2 MORE LINES AND
NOTREQ  0ECD 60 RTS                   ;RETURN

                 ;INCREMENT POINTER SUBROUTINE
                 ;INCREMENTS TEXT EDITOR POINTERS
INCPNT  0ECE E6 INC 00       TELSB    ;INCREMENT LSB POINTER
        0ED0 D0 BNE 0ED4     INCPT1   ;RETURN IF NOT END OF PAGE
        0ED2 E6 INC 01       TEMSB    ;INCREMENT MSB POINTER
INCPT1  0ED4 60 RTS                   ;RETURN
                 ;CHANGE DISASSEMBLE ADDRESS SUBROUTINE
                 ;ALTERS POINTERS FOR NEXT DISASSEMBLED LINE
ADDRIN  0ED5 48 PHA
        0ED6 A2 LDX #0F               ;TAB OVER 15 SPACES
        0ED8 20 JSR 0F7A     SPACES
        ØFCA C9 CMP #2Ø
        ØFCC FØ BEQ ØFD9   DEFALT   ;IF SPACE GO TO DEFALT
        ØFCE 2Ø JSR E97A   OUTPUT
        ØFD1 2Ø JSR ØECE   INCPNT   ;IF NOT PRINT IT &
        ØFD4 CA DEX                 ;GET THE NEXT CHAR
        ØFD5 DØ BNE ØFC8   GECHAR
        ØFD7 FØ BEQ ØFE2   LEAVE    ;IF 6 CHARS PRINTED, RETURN
DEFALT  ØFD9 2Ø JSR E83E   BLANK
        ØFDC CA DEX
        ØFDD DØ BNE ØFD9   DEFALT   ;OTHERWISE TAB TO FILL
        ØFDF 2Ø JSR ØECE   INCPNT   ;IN ALL 6 ALLOCATED SPACES
LEAVE   ØFE2 6Ø RTS                 ;AND RETURN

                    ;MESSAGE SUBROUTINE
                    ;OUTPUTS ASCII MESSAGE FROM TABLE
MESS    ØFE3 B5 LDA Ø6,X   MTABLE   ;GET CHARACTER
        ØFE5 48 PHA
        ØFE6 29 AND #7F             ;MASK BIT 8
        ØFE8 2Ø JSR E97A   OUTPUT   ;PRINT CHARACTER
        ØFEB E8 INX
        ØFEC 68 PLA                 ;IF TABLE VALUE JUST PRINTED
        ØFED 1Ø BPL ØFE3   MESS     ;WAS NOT NEGATIVE,
        ØFEF 6Ø RTS                 ;GET NEXT CHARACTER

                    ;PATCH SUBROUTINES
                    ;FIXES TO OTHER ROUTINES TO AVOID CHANGING BRANCHES
PATCH1  ØFFØ AØ LDY #ØØ             ;RESET Y TO ZERO
        ØFF2 2Ø JSR ØFC6   GETLBL   ;PRINT ARGUMENT LABEL
        ØFF5 6Ø RTS                 ;OR SPACES AND RETURN
PATCH2  ØFF6 C6 DEC Ø2     LCOUNT   ;DECREMENT LINE COUNTER SO
        ØFF8 C6 DEC Ø2     LCOUNT   ;TITLE & * LINE WILL
        ØFFA C6 DEC Ø2     LCOUNT   ;NOT COUNT FOR PAGING
        ØFFC 6Ø RTS

                    ;MESSAGE TABLE
                    ;USED BY MESSAGE SUBROUTINE
    <M>=ØØØ6 2Ø 2Ø 2Ø 2Ø   - - - -  ;TAB SPACES
    <M>=ØØØA 5Ø 41 47 45   P A G E  ;PAGE HEADING
    <M>=ØØØE AØ 2Ø 2Ø 2Ø   - - - -  ;TAB SPACES
    <M>=ØØ12 3C 4D 3E BD   < M > =  ;ASCII TABLE PREFIX

                    ;END OF LISTING
```

Figure 5

MERGE COMMENTS WITH DISASSEMBLY - OPERATING INSTRUCTIONS
****

1. INITIALIZE THE TEXT EDITOR WITH THE "E" COMMAND. DEFINE THE LIMITS TO START JUST ABOVE THE TOP OF YOUR SUBJECT PROGRAM AND END JUST BELOW THE START OF MERGE.

2. THE FIRST LINE ENTERED INTO THE TEXT EDITOR MUST BE A "TITLE LINE" CONTAINING THE NAME OF YOUR PROGRAM. THIS IS THE LINE THAT WILL BE PRINTED AT THE TOP OF SHEETS 2 AND UP OF YOUR PROGRAM LISTING. SEE BELOW FOR "TITLE LINE" DIRECTIONS.

3. ALL LINES ENTERED INTO THE TEXT EDITOR WILL FALL INTO ONE OF THE FOLLOWING SIX CATAGORIES. INPUT EACH LINE INTO THE EDITOR ACCORDING TO THE FOLLOWING INSTRUCTIONS:

A. BLANK LINE
   1) TYPE IN AS "CONTROL S" "RETURN".
   2) DO NOT USE "SPACE" "RETURN".
   3) PROVIDES A LINE FEED WHEN ENCOUNTERED.

B. TITLE LINE
   1) LINE MUST START WITH A "CONTROL T" AND END WITH "RETURN".
   2) TABS PRINTER TO PRINT THE LINE STARTING AT THE 16TH CHARACTER SPACE. THE "CONTROL T" IS IGNORED.
   3) LENGTH OF LINE IS LIMITED BY PRINTER WIDTH.

C. COMMENT LINE
   1) LINE MUST START WITH A SEMICOLON (;) AND END WITH "RETURN".
   2) TABS PRINTER TO PRINT THE LINE STARTING AT THE 16TH CHARACTER SPACE. THE SEMICOLON IS PRINTED.
   3) LENGTH OF LINE IS LIMITED BY THE PRINTER WIDTH.

D. ASTRIX LINE
   1) LINE MUST START WITH AN ASTRIX (*).
   2) DEFINES THE ADDRESS FOR THE NEXT INSTRUCTION TO BE LISTED BY A "DISASSEMBLED PROGRAM LINE".
   3) TYPE IN AS "*0200" "RETURN", FOR EXAMPLE, WITH "0200" BEING THE DESIRED ADDRESS IN THE SUBJECT PROGRAM.
   4) TABS PRINTER TO PRINT THE LINE AS "*=0200" STARTING AT THE 16TH CHARACTER SPACE.
   5) MUST BE ENTERED AHEAD OF THE FIRST "DISASSEMBLED PROGRAM LINE" IN EACH SECTION OF THE TEXT EDITOR TO DEFINE THE START ADDRESS.
   6) MERGE WILL THEN CONTINUE TO DISASSEMBLE CONSECUTIVE PROGRAM STEPS EACH TIME A "DISASSEMBLED PROGRAM LINE" FORMAT IS CALLED FOR UNTILL ANOTHER "ASTRIX LINE" IS ENCOUNTERED.
   7) AN "ASTRIX LINE" MAY BE INSERTED AT ANY TIME TO JUMP TO A NEW LOCATION OR TO SKIP GAPS IN THE SUBJECT PROGRAM.

E. ASCII TABLE LINE
   1) LINE MUST START WITH A "CONTROL A" AND END WITH A "RETURN".
   2) THE FIRST LINE OF EACH TABLE MUST FOLLOW THE "CONTROL A" WITH THE SEQUENCE "*0006", FOR EXAMPLE, WITH THE "0006" BEING THE ADDRESS OF THE START OF THE MESSAGE TABLE IN THE SUBJECT PROGRAM. SUCCESSIVE LINES STARTING WITH ONLY "CONTROL A" WILL LIST THE NEXT CONSECUTIVE FOUR BYTES OF THE TABLE.
   3) THE ALPHANUMERIC CHARACTERS FOR THAT LINE OF THE TABLE ARE THEN ENTERED AS "ABCD", FOR EXAMPLE. THERE MUST BE 4 CHARACTERS PER LINE ENTERED. USE SPACES IF REQUIRED TO FILL OUT THE LAST LINE.
   4) IF THERE ARE NO COMMENTS TO BE ADDED TO THIS LINE, TYPE "RETURN" AND GO ON TO THE NEXT LINE.
   5) IF THERE ARE COMMENTS FOR THIS LINE, BEGIN TYPING THEM IMMEDIATLY AFTER THE 4TH ALPHANUMERIC CHARACTER HAS BEEN ENTERED. THEY WILL BE RUN TOGETHER

IN THE TEXT EDITOR BUT MERGE WILL SEPARATE THEM AND ADD A SEMICOLON WHEN PRINTING. END THE LINE WITH "RETURN".

6) LENGTH OF COMMENTS LIMITED BY PRINTER WIDTH.

F. DISASSEMBLED PROGRAM LINE

1) IF A LINE STARTS WITH ANY THING OTHER THAN A "CONTROL S", "CONTROL T", "CONTROL A", ASTRIX OR SEMICOLON, THE LINE WILL BE TREATED AS A DISASSEMBLED PROGRAM LINE.

2) IF THERE ARE NO LABELS OR COMMENTS FOR THIS LINE, TYPE "SPACE" "SPACE" "RETURN" AND GO ON TO THE NEXT LINE.

3) WHEN THEY ARE DESIRED, A PROGRAM LABEL, AN ARGUMENT LABEL AND COMMENTS MAY BE ENTERED FOR A LINE. THEY ARE ALWAYS ENTERED IN THAT ORDER. A LINE NEED NOT HAVE ALL OF THEM, JUST THE ONES YOU DESIRE.

4) IF THERE IS NO PROGRAM LABEL FOR THIS LINE, TYPE ONE "SPACE". THIS WILL DEFAULT TO SIX SPACES WHEN PRINTING. IF THERE IS A PROGRAM LABEL FOR THIS LINE, TYPE IT IN. A LABEL MAY NOT BE MORE THAN SIX CHARACTERS LONG. IF IT IS LESS THAN 6 CHARACTERS LONG, TYPE THE LABEL FOLLOWED BY ONLY ONE "SPACE". THE REQUIRED SPACES WILL BE PUT IN WHEN PRINTING.

5) NEXT, FOLLOW THE IDENTICAL INSTRUCTIONS FOR THE ARGUMENT LABEL (IF ANY). IF THE PROGRAM LABEL WAS THE FULL 6 CHARACTERS LONG, THE 2 LABELS WILL BE RUN TOGETHER IN THE EDITOR (EX: "PLABELALABEL"). THE PRINT ROUTINE WILL SEPARATE THEM.

6) IF THERE ARE NO COMMENTS FOR THIS LINE, TYPE "RETURN" AND GO ON TO THE NEXT LINE.

7) IF THERE ARE COMMENTS TO BE ADDED TO THIS LINE, BEGIN TYPING THEM IMMEDIATLY AFTER THE ARGUMENT LABEL OR DEFAULT SPACE. IF THE ARGUMENT LABEL WAS THE FULL 6 CHARACTERS LONG, THE LABEL AND COMMENTS WILL BE RUN TOGETHER IN THE EDITOR (EXAMPLE: "ALABELCOMMENTS..."). THE PRINT ROUTINE WILL SEPARATE THEM AND WILL ALSO PREFIX THE COMMENTS WITH A SEMICOLON. END THE LINE WITH "RETURN".

8) LENGTH OF COMMENTS IS LIMITED BY PRINTER WIDTH.

4. WHEN ALL OF THE LINES HAVE BEEN ENTERED OR YOU RUN OUT OF MEMORY IN THE EDITOR, TERMINATE THE TEXT EDITOR INPUT WITH THE NORMAL "RETURN" "RETURN" SEQUENCE. IF YOU HAVE RUN OUT OF MEMORY, RE-ENTER THE TEXT EDITOR WITH THE "T" COMMAND AND THEN GO TO THE BOTTOM WITH THE "B" COMMAND. DELETE THE LAST TWO OR THREE LINES WITH THE "K" COMMAND TO OBTAIN A LITTLE EDITING ROOM IF YOU SHOULD NEED IT LATER.

5. EXIT THE TEXT EDITOR WITH THE "Q" COMMAND.

6. LOAD MERGE AND YOUR SUJECT PROGRAM INTO RAM.

7. START MERGE WITH THE "F2" (!) KEY AND RESPOND TO "FROM=" WITH THE 4 DIGIT START ADDRESS THAT YOUR TEXT EDITOR HAS WAS SET TO FOLLOWED BY "RETURN". YOUR SUBJECT PROGRAM AND YOUR LABELS AND COMMENTS WILL BE MERGED AND FORMATTED AS THEY ARE PRINTED OUT. WHEN THE END OF THE DATA IN THE TEXT EDITOR IS REACHED, MERGE WILL EXIT TO THE AIM MONITOR. MAKE A RECORD OF THE CONTENTS OF 0002H AND 0003H AT THIS TIME. YOU WILL NEED TO KNOW IT LATER.

8. MAKE ANY CORRECTIONS TO THE DATA IN THE TEXT EDITOR BY RE-ENTERING THE EDITOR WITH THE "T" COMMAND. USE THE STANDARD AIM EDITING COMMANDS AND THE SAME DATA INPUT RULES TO FIX UP WHATEVER LINES NEED CHANGED. PRINT ANOTHER LISTING FOLLOWING THE INSTRUCTIONS IN STEP 7 AGAIN. WHEN VERIFIED CORRECT, SAVE THE TEXT EDITOR DATA ON TAPE USING THE "L" COMMAND.

9. NOTE THAT IF YOU LIST YOUR TEXT EDITOR DATA ON THE AIM THERMAL PRINTER, IT WILL PRINT AN "A", "S" OR "T" WHEN IT FINDS A "CONTROL A", "CONTROL S" OR "CONTROL T" RESPECTIVELY IN THE TEXT EDITOR. THIS WILL NOT PRINT DURING THE RUNNING OF MERGE NOR WILL IT PRINT IF YOU LIST THE TEXT EDITOR DATA TO AN EXTERNAL PRINTER.

10. IF YOUR LABELS AND COMMENTS ALL FIT INTO YOUR TEXT EDITOR AT ONE TIME AND THE LISTING PRODUCED IS COMPLETE AND TO YOUR SATISFACTION, YOU ARE DONE. SAVE MERGE, YOUR SUBJECT PROGRAM AND THE LABEL AND COMMENT DATA ON TAPE IN CONSUTIVE SAVES TO PERMIT LISTING IT IN THE FUTURE. YOU MUST HAVE ALL THREE OF THEM IN RAM AT ONCE TO RUN A LISTING. MERGE DOESN'T COMBINE THE PROGRAM AND COMMENTS IN RAM, IT JUST PRINTS THEM OUT TOGETHER.

11. IF YOUR TEXT EDITOR RAN OUT OF CAPACITY BEFORE YOU GOT ALL THE WAY THROUGH YOUR SUBJECT PROGRAM DOCUMENTATION, ADDITIONAL SECTIONS OF DATA WILL BE REQUIRED TO COMPLETE THE LISTING. THE INSTRUCTIONS FOLLOWING WILL BE APPLICABLE FOR SECTIONS 2, 3, 4, ETC. OF THE LABEL AND COMMENT INPUT AND PRINTING:

A. MAKE CERTAIN THAT YOU HAVE A GOOD SAVE ON TAPE OF THE DATA FOR THE LAST SECTION THAT IS IN THE TEXT EDITOR. INITIALIZE YOUR TEXT EDITOR AGAIN AND SET THE UPPER AND LOWER LIMITS THE SAME AS THE LAST SECTION WAS. YOU ARE NOW READY TO INPUT ANOTHER SECTION OF DATA.

B. THE FIRST LINE OF EACH SECTION MUST BE THE TITLE OF THE PROGRAM IN A "TITLE LINE" FORMAT. THIS MUST READ THE SAME AS THE TITLE LINE IN THE FIRST SECTION.

C. THE SECOND LINE OF EACH SECTION MUST BE AN "ASTRIX LINE" CONTAINING THE ADDRESS OF THE NEXT INSTRUCTION TO BE DISASSEMBLED. THE CONTINUITY IS LOST EACH TIME MERGE EXITS TO THE MONITOR AND IT MUST BE RESTORED UPON RE-ENTRY BY THE DATA IN THE EDITOR.

D. INPUT LABEL AND COMMENT DATA INTO THE TEXT EDITOR STARTING JUST AFTER THE LAST SECTION LEFT OFF. USE

THE SAME INPUT RULES AS OUTLINED IN INSTRUCTION 3 AND DO INSTRUCTIONS 4 AND 5.

E. RE-ENTER MERGE WITH THE "F1" ([) COMMAND. RESPOND TO THE "FROM=" PROMPT THE SAME AS BEFORE(INSTRUCTION 7). MERGE WILL AGAIN PRINT UNTILL IT RUNS OUT OF DATA IN THE TEXT EDITOR. WHEN IT EXITS TO THE AIM MONITOR, MAKE A RECORD OF THE CONTENTS OF 0002 AND 0003 AGAIN.

F. MAKE ANY CORRECTIONS NEEDED TO THIS SECTION USING THE TEXT EDITOR COMMANDS AGAIN.

G. REGISTERS 0002 AND 0003 KEEP TRACK OF THE LINE COUNT AND THE PAGE COUNT FOR THE PAGE SUBROUTINE. WHEN MERGE IS ENTERED WITH "F2" (]), THEY ARE INITIALIZED TO START FROM PAGE 1, LINE 1. AS THE TEXT IS PRINTED, THEY ARE INCREMENTED. WHEN EDITING AND REPRINTING THE THE FIRST SECTION OF DATA AGAIN USING "F2" (]), THE REGISTERS WOULD BE RESET AND REINCREMENTED EACH TIME.

WHEN YOU RE-ENTER MERGE WITH THE "F1" ([) KEY, THESE REGISTERS ARE NOT RESET. THEY START WHERE THEY LEFT OFF AND CONTINUE TO INCREMENT. THIS IS DONE TO PRESERVE THE PAGING WHEN DOING SEQUENCIAL RUNS USING MULTIPLE DATA SECTIONS.

HOWEVER, WHEN RUNNING THE SAME SECTION OVER A FEW TIMES DURING THE COURSE OF CORRECTING AND RUNNING SUBSIQUENT PROOF COPIES, THESE REGISTERS GET ALL OUT OF SYNC. FOR THAT REASON YOU MUST CHECK THE VALUES OF THESE REGISTERS AFTER EACH RUN AND RECORD THEM. TO GET THE PAGING BACK IN SYNC, MANUALLY SET 0002 AND 0003 TO THE VALUE THAT THEY WERE AT AFTER THE LAST SECTION WAS FINALIZED AND RUN. FOR EXAMPLE, LETS SAY YOU ARE ON THE SECOND SECTION. THE FIRST SECTION ENDED UP WITH 2A IN 0002 AND 02 IN 0003. BEFORE YOU RERUN ANOTHER PROOF COPY OF SECTION 2, MANUALLY SET 0002 TO 2A AND 00003 TO 02. THEN RERUN WITH THE "F1" ([) KEY.

IF YOU SHOULD LOSE TRACK OF THE REGISTER VALUES, SAVE THE CURRENT TEXT EDITOR DATA ON TAPE. TURN OFF THE AIM THERMAL PRINTER. RELOAD THE FIRST DATA SECTION FROM TAPE. RUN MERGE, WITH THE OUTPUT GOING TO THE DISPLAY ONLY, UP TO THE POINT WHERE YOU ARE. KEEP TRACK OF THE REGISTERS AS YOU GO.
ONCE YOU HAVE GOT THE PRESENT SECTION CORRECT AND HAVE RUN A GOOD PROOF, SAVE IT ON TAPE. IF THERE IS MORE TO DO, JUMP BACK TO INSTRUCTION 11-A AND DO THE NEXT SECTION. OTHERWISE, GO ON TO 11-H.

H. ALL THAT REMAINS TO DO NOW IS TO CUT OUT THE EXTRANEOUS LINES THAT ARE PRINTED EACH TIME MERGE IS RE-ENTERED WITH THE "F1" ([) KEY. THERE WILL BE A "TITLE LINE" AND AN "ASTRIX LINE" PRINTED ALONG WITH A FEW BLANK LINES THAT YOU DON'T WANT. CUT THE LISTING JUST ABOVE THE FIRST GOOD LINE OF THE SECTION. MOVE THE LISTING UP SO THE FIRST GOOD LINE IS JUST AFTER THE LAST GOOD LINE AND TAPE OR PASTE THE SECTIONED LISTING BACK TOGETHER. IF YOU HAVE DONE IT CORRECTLY, THE CONTINUITY OF THE PAGING WILL HAVE BEEN RESTORED. MARK OFF THE CUT LINES FOR ALL OF THE PAGES IN THE LISTING AND CUT THEM. I USE THE CARDBOARD BACKING FROM AN 8 1/2" X 11" TABLET AS A TEMPLATE WHEN MARKING THE PAGES OFF. ONCE YOU GET THE TOP MARGIN SET ON THE FIRST PAGE, THE REST OF THE PAGES FALL RIGHT INTO PLACE.

13. WHILE THIS HAS PROBABLY BEEN CONFUSING TO READ THROUGH, THE ENTIRE OPERATION IS EASY TO GET USED TO. READ THROUGH IT AGAIN WHILE DOING THE STEP BY STEP DOCUMENTATION FOR YOUR FAVORITE PROGRAM. DISREGARD ANY QUOTATION MARKS (") SHOWN IN THESE OPERATING INSTRUCTIONS. THEY ARE SHOWN HERE TO EMPHASIZE OR SEPARATE THE VARIOUS FORMATS TO USE. THEY ARE NOT A PART OF ANY COMMAND OR ENTRY PROCEDURE. ONCE YOU HAVE USED MERGE, IT WILL ALL BECOME CLEAR TO YOU. AFTER YOU HAVE MADE YOUR FINAL LISTING, WRITE UP A SHORT COVER STORY AND MAIL IT WITH YOUR PROGRAM TO YOUR FAVORITE PUBLISHER.

Introducing the Atari 800

Perhaps you have heard about the Atari, or maybe even have seen it's picture in the Sears Roebuck Catalog! Here are the basic facts about this latest addition to the expanding 6502 microcomputer world.

William L. Colsher

The model 800 computer is Atari's "top of the line" personal computer. Included in the $999.95 list price are the Atari 410 cassette program recorder, an 8K BASIC in a removable ROM cartridge, an Educational System ROM cartridge, the 332 page Atari BASIC book by Albrecht, Finkel and Brown, and several other small manuals. Though it was not yet ready when my system was delivered, buyers should now be getting a short CAI course called "An Invitation to Programming". This, along with the Atari BASIC book should bring the beginner up to speed pretty quicklv.

The model 400 includes everything except the program recorder and Educational System cartridge for $599.95. It is functionally identical with the 800 but with the following differences: the keyboard is a flat 'elastomer' type keyboard, the memory can be expanded only to 16K (and that requires a visit to the shop) and the only peripheral you can use (other than game controllers) is the 410 program recorder.

Atari has several peripherals available now including the 810 disk drive at $699.95, the 820 printer and $599.95 and an assortment of game controllers. Announced but with no prices as yet are the 825 printer (apparently a Centronics 730), the 830 modem which looks just like a Novation CAT modem, and the 850 Interface Module which features four RS232 ports and a Centronics port so you can use the 825 printer. One of the RS 232 ports also supports 20 mA current loop so you can use it with a teletype.

Color graphics are one of the Atari's strong points. There are eight different modes each of which can operate in several different ways— it is complicated but *very* flexible. Graphic resolution is from 39 x 20 all the way up to 320 x 192. In all modes but the last you can have up to 4 colors on the screen at a time. (Three colors for the graphics points plus a background color.) In addition to the color, you can also control something called luminance. This is roughly like an artist adding white to another color. For example, you can get red in any shade from very pale pink to a dark, bloody shade (great for D&D).

The Atari computers alos include an excellent sound system: four separate voices with individual tone and volume control. The frequencies range from about an octave below middle C to just barely audible at the high end.

Since it had to fit into 8K of ROM, Atari BASIC is smallish but quite adequate. The only major things missing are the string intrinsics and some of the convenience items like automatic line numbering when you are entering programs and line renumbering.

The lack of convenience features is pretty well made up for by Atari's *super* screen editor. Basically, if it's on the screen and you change it, it's changed in memory. To make a program change just LIST the line and stick in the changes on the screen. That's all there is to it. Various function keys make it easy to insert and delete spaces and even entire lines. Ofcourse ther are the usual cursor control keys for moving around on the screen.

While we are on the subject of the keyboard, I'll point out that someone inside Atari's development team seems to have thought of just about everything. Included are all the usual keys as well as CTRL and ESC. You have control of the keyboard graphics symbols, upper and lower case, and normal and reverse video. The only thing missing form the 800's keyboard are the graphic symbols which are printed on the 400's. Perhaps someone will come out with a set of stickers (are you there, Atari?).

In terms of software, Atari seems to be doing pretty well. There is a definite emphasis on education with 17 packages in a wide range of subjects. ROM cartridges are also big with 11 available. I've mentioned two that come with the 800; also in the catalog are an Assembler, Chess, Star Raiders (an absolutely fantastic game) and a number of other. If all the games are as great as Star Raiders, then they are well worth the prices Atari charges— I've easily saved $59.95 in quarters (I'm an arcade game freak).

Other companies with software for the Atari machines are The Code Works which publishes IRIDIS (the first issue is available now) and Image Computer Products. Sears, Roebuck Co. has eight cassettes listed in their current catalog as well. (The idea of 'Sears Software' is a bit of a shock at first!)

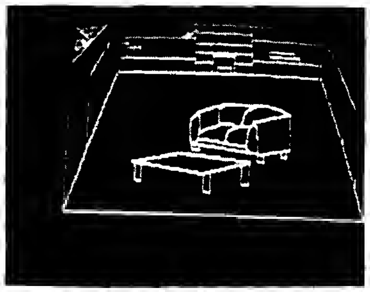
I think the Atari machines are a good buy with quite a future ahead. The 800 is as versatile as any other machine in its price range, comparable to the Apple II and easily outdistancing the TRS-80 (I have a 48K dual disk "—80" as well). With companies like Sears getting into the personal computer game, we are probably in for something of a revolution. And Atari should be in the front lines.

NEW APPLE II / APPLE II PLUS SOFTWARE FROM ON– LINE SYSTEMS

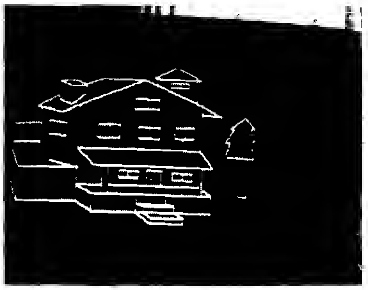
TIRED OF BUYING GAMES THAT BECOME BORING AFTER A FEW HOURS OF PLAY? ON–LINE SYTEMS IS DEDICATED TO DELIVERING SERIOUS SOFTWARE FOR THE DISCRIMINATING GAMESMAN. THESE PRODUCTS HAVE BEEN SIX MONTHS IN DEVELOPMENT AND PROVIDE THE QUALITY AND SPEED POSSIBLE ONLY THROUGH MACHINE LANGUAGE!

ALL NEW

HI-RES ADVENTURE (''MYSTERY HOUSE'')



What is an adventure game? According to the dictionary, an adventure is a hazardous or daring enterprise; an exciting experience; to risk, hazard, to venture on. One who goes on an adventure is a venturer. A seeker of fortune in daring enterprises; a speculator. In essence, an adventure game is a fantasy world where you are transported, via your own computer. You are the key character of the fantasy as you travel through a land the likes of which you will find in books that take you, through your imagination, to the world it is creating.



Through the use of over a hundred Hi-Res pictures you play and see your adventure. You communicate with HI-RES ADVENTURE in plain english (it understands over 300 words!) All rooms of this spooky old house appear in full Hi-Res Graphics complete with objects you can get, carry, throw, drop, or ?.

In this particular HI-RES ADVENTURE game, you are transported to the front yard of a large, old victorian house. When you enter the house you are pulled into the mystery, murder, and intrigue and can not leave until you solve the puzzles. Your friends are being murdered one by one. You must find out why, and who the killer is. Be careful, because the killer may find you! As you explore the house there are puzzles to be solved and hazards to overcome. The secret passage-way may lead you to the answer.

ALSO NEW FROM ON–LINE SYSTEMS

SKEET/TRAP have become Olympic shooting sports and and obsession among Scatter-gunners all over the world. These games are the All-American although they have become international.

SKEETSHOOT allows one to five shotgunners to test their marksmanship as they fire from the eight prescribed positions on an official NSSA skeet firing range. Each position provides a new perspective of the field with the pigeons travelling at different angles. At each position a pigeon is launched from one side of the field and then the other. At certain positions, pigeons are launched from both sides of the field simultaneously. This is a true game of skill, simulating skeet shooting down to the last detail.

TRAPSHOOT allows one to five shotgunners to test their markmanship. The trap firing range has five positions where the one to five players shoot from. Each player is at a different location on the field. The challenge is to shoot pigeons out of the sky which launch at random trajectories. The challenge is to hit the pigeons while they are still in gun range.

SKEETSHOOT and TRAPSHOOT both allow you to control the size and speed of the pigeons and the width of your shotgun spray. Realistic sound-effects and HI-RES animation combine to make this simulation unparrelled for the APPLE.

DEALER INQUIRIES INVITED!

ALL SOFTWARE SHIPPED SAME DAY.
PHONE ORDER: (805) 522-8772

ON–LINE SYSTEMS, 772 NO HOLBROOK, SIMI, CA 93065

SEND TO: ______________________________

____

		QUANTITY	TOTAL
Hi-Res Adventure/Disk	$24.95		
Skeetshoot/Cassette	$14.95		
Trapshoot/Cassette	$ 9.95		
Skeetshoot/Disk	$19.95		
Trapshoot/Disk	$14.95		
Hi-Res Adventures & Skeetshoot/Disk	$37.50		
		Subtotal	
Payment: Check________		6% tax (Calif.)	
Master Chg/Visa #________		Shipping	$1.00
Expires:________		Total	

APPLE II Integer BASIC Program List by Page

A number of programs have been written which solve the Apple's problem of examining a program on the display. This version permits the user to simply view his program page-by-page.

Dave Partyka

If you own an Apple II, I'm sure you feel there could be a better way to list a program. The way it is now you either list the whole program and watch it go by faster than you can read it, or you list it by line numbers. When you list it by line numbers, you may get two lines or you may get more lines than will fit on the screen.

Using the assembler program listed, and the integer basic of the Apple II, you can list your integer basic programs one page (screen) at a time with a page number at the bottom of each. Pressing just about any key (except B, P, or S) will clear the screen and display the next page adding one to the page number. By pressing keys you display your program a page at a time, with no more two lines here or too many lines there.

You are probably wondering why you can't use the B, P, or S keys. These are special function keys. The B key (for beginning) will clear the screen and display your program from the first page. This comes in handy when you're in the middle or near the end of the display and you want to see some subroutines or anything else at the beginning. Just press the B key and you are at the beginning, ready to start over.

The next key, P (for page) will clear the screen and start displaying your program stopping at the page number you keyed in. For example, if you are at page 25 and you want to back up 2 pages, you press P0023. P will clear the screen and the APPLE

```
APPLE II
INTEGER BASIC
PROGRAM LIST BY PAGE

300   A9 22      LDA #22    LOAD BEGINNING
302   85 36      STA  36    ADDRESS OF MAIN
304   A9 03      LDA #03    PROGRAM IN USER
306   85 37      STA  37    OUTPUT LOCATIONS.
308   20 E6 03   JSR  3E6   LOAD HIGH VALUES.
30B   A9 00      LDA #00    MOVE ZEROS TO
30D   8D F4 03   STA  3F4   PAGE COUNT
310   8D F5 03   STA  3F5   LOCATIONS.
313   20 58 FC   JSR  FC58  CLEAR SCREEN.
316   20 4B E0   JSR  E04B  START BASIC LIST.


319   20 96 03   JSR  396   ADD 1 TO PAGE #.
31C   20 E6 03   JSR  3E6   LOAD PAGE HOLD WITH FF.
31F   4C 03 E0   JMP  E003  RETURN TO BASIC CONTROL.
322   48         PHA        SAVE ACCUM. AND IND-Y
323   98         TYA        VALUES BEFORE PRINTING
324   48         PHA        ON THE SCREEN.
325   A5 28      LDA  28    CHECK SCREEN ADDRESS
327   45 29      EOR  29    FOR 07 DO THE
329   C9 D7      CMP #D7    24TH LINE.
32B   D0 54      BNE .381   IF NOT = BRANCH.
32D   20 96 03   JSR  396   ADD 1 TO PAGE #.


330   AD F6 03   LDA  3F6   CHECK PAGE HOLD,
333   C9 FF      CMP #FF    IF = FF THEN THE P
335   F0 19      BEQ  350   KEY WASN'T PRESSED.
337   AD F4 03   LDA  3F4   COMPARE PAGE #
33A   CD F6 03   CMP  3F6   WITH PAGE HOLD,
33D   D0 08      BNE  347   IF EQUAL,
33F   AD F5 03   LDA  3F5   BRANCH TO THE
```

will beep as you key in the four digits. You have to enter four digits so the leading zeros are necessary. After the last digit is pressed, your program will be displayed from the beginning, stopping at page 23. This is faster than pressing the B key and other ones until you get to page 23.

The last key, S (for Stop) gets you out of the list program and back to the APPLE II basic. This key is used when you find a place in your program where you want to add or delete a line. If you don't press the S key and you try to do anything, as soon as you press a key the next page will be displayed.

There are two ways to activate this program. From monitor press CTRL Y then the RETURN key or from basic type CALL 1016 then press the RETURN key. As long as you don't use the area from hex 300 to 3FF, this program will remain in memory. Once the list program is activated, it is entered only when the screen display reaches the bottom of the screen. If the end of your program ends anywhere but the bottom of the screen, the Apple II will return to basic but the list program will still be activated. To deactivate the list program, type CALL 1016 then press the RETURN key, then press the S key for stop, or press the RETURN key to skip to the bottom of the page and press the S key to stop.

If you ran a basic program and the list program is still activated, then the results you get will depend on your program. Some programs won't be affected at all while others will stop if it has a display that reaches the bottom of the screen. Pressing a key will start the program again. Other programs might be able to make use of this assembler routine by stopping the display at the bottom of the screen.

Using this assembler program, you'll find it easier to de-bug your programs or just follow the flow of any program. μ

Dave Partyka works as a programer on an IBM 3031 OS system for the May dept. store company. He has been programming for 3 years, and he has been an operator for 4 years prior to that. Before he began work at the May company he served 4 years in the US Navy where he worked in data processing.

```
342   CD F7 03   CMP  3F7    LOOP ROUTINE
345   FO 06      BEQ  34D    ELSE
347   20 58 FC   JSR  FC58   CLEAR SCREEN

34A   4C 81 03   JMP  381   CONTINUE PRINTING.
34D   20 E6 03   JSR  3E6   LOAD PAGE HOLD WITH FF.
350   2C 00 CO   BIT  C000  LOOP UNTIL A
353   10 FB      BPL  350   KEY IS PRESSED.
355   AD 00 CO   LDA  C000  WHEN KEY IS PRESSED
358   8D 10 CO   STA  C010  CLEAR KEY STROBE
35B   C9 D3      CMP #D3    AND COMPARE FOR S.
35D   DO OB      BNE  36A   IF NOT = BRANCH.
35F   A9 FO      LDA #FO    IF S STORE
361   85 36      STA  36    NORMAL ADDRESS

363   A9 FD      LDA #FD    IN THE USER
365   85 37      STA  37    OUTPUT LOCATIONS.
367   4C 03 EO   JMP  E003  RETURN TO BASIC CONTROL.
36A   C9 C2      CMP #C2    B KEY PRESSED?
36C   FO 9A      BEQ  308   IF YES BRANCH.
36E   C9 DO      CMP #DO    F KEY PRESSED?
370   DO OC      BNE  37E   IF NO BRANCH.
372   A2 00      LDX #00    IF YES THEN GET
374   20 CF 03   JSR  3CF   2 DIGITS OF PAGE #.
377   E8         INX        UP INDEX AND

378   20 D2 03   JSR  3D2   GET NEXT TWO DIGITS.
37B   4C OB 03   JMP  30B   JUMP TO ZERO PAGE #.
37E   20 58 FC   JSR  FC58  CLEAR SCREEN.
381   68         PLA        GET ACCUM. AND IND-Y
382   A8         TAY        FROM THE STACK
383   68         PLA        AND JUMP TO THE
384   4C FO FD   JMP  FDFO  DISPLAY ROUTINE.
387   A8         TAY        SAVE ACCUM. AND
388   29 OF      AND #OF    CONVERT LOW ORDER
38A   09 BO      ORA #BO    BYTE TO DECIMAL AND

38C   9D F4 07   STA  7F4,X   PRINT PAGE #.
38F   98         TYA          GET ACCUM. AND
390   6A         ROR          ROTATE
391   6A         ROR          HIGH ORDER
392   6A         ROR          BYTE TO THE
393   6A         ROR          LOW ORDER
394   CA         DEX          BYTE AND
395   60         RTS          RETURN.
396   F8         SED          SET DECIMAL MODE.
397   18         CLC          CLEAR CARRY FLAG.

398   AD F5 03   LDA  3F5     ADD
39B   69 01      ADC #01      1
39D   8D F5 03   STA  3F5     TO
3A0   AD F4 03   LDA  3F4     THE
3A3   69 00      ADC #00      PAGE
```

```
3A5   8D F4 03   STA  3F4     NUMBER.
3A8   D8         CLD          CLEAR DECIMAL MODE.
3A9   A2 03      LDX #03      SET IND-X.
3AB   AD F5 03   LDA  3F5     GET PAGE # LOW.
3AE   20 87 03   JSR  387     PRINT 1ST DIGIT.
3B1   20 87 03   JSR  387     PRINT 2ND DIGIT.
3B4   AD F4 03   LDA  3F4     GET PAGE # HIGH.
3B7   20 87 03   JSR  387     PRINT 3RD DIGIT.

3BA   20 87 03   JSR  387     PRINT 4TH DIGIT.
3BD   60         RTS          RETURN.
3BE   2C 00 C0   BIT  C000    LOOP UNTIL A
3C1   10 FB      BPL  3BE     KEY IS PRESSED.
3C3   20 DD FB   JSR  FBDD    RING BELL
3C6   AD 00 C0   LDA  C000    GET KEY
3C9   8D 10 C0   STA  C010    CLEAR STROBE
3CC   29 0F      AND #0F      DROP HIGH ORDER
3CE   60         RTS          HALF AND RETURN.
3CF   20 58 FC   JSR  FC58    CLEAR SCREEN.

3D2   20 BE 03   JSR  3BE     GET PAGE #.
3D5   0A         ASL          SHIFT LOW ORDER
3D6   0A         ASL          HALF TO THE
3D7   0A         ASL          HIGH ORDER
3D8   0A         ASL          HALF.
3D9   9D F6 03   STA  3F6,X   STORE IN PAGE HOLD.
3DC   20 BE 03   JSR  3BE     GET NEXT NUMBER.
3DF   5D F6 03   EOR  3F6,X   COMBINE WITH
3E2   9D F6 03   STA  3F6,X   PREVIOUS # AND STORE
3E5   60         RTS          IN PAGE HOLD, RETURN.

3E6   A9 FF      LDA #FF      PUT HIGH VALUES
3E8   8D F6 03   STA  3F6     IN PAGE HOLD
3EB   8D F7 03   STA  3F7     LOCATIONS THEN
3EE   60         RTS          RETURN.
3EF   00         BRK
3F0   00         BRK
3F1   00         BRK
3F2   00         BRK
3F3   00         BRK
3F4   00         BRK          PAGE # HIGH

3F5   00         BRK          PAGE # LOW
3F6   00         BRK          PAGE HOLD HIGH
3F7   00         BRK          PAGE HOLD LOW
3F8   4C 00 03   JMP  300     CTRL-Y ENTERS HERE
3FB   00         BRK
3FC   00         BRK
3FD   00         BRK
3FE   00         BRK
3FF   00         BRK
```

Ask for Instant Software at a computer store near you.

Alabama

Anderson Computers
3156 University Dr., Hunlsville

Computerland ot Huntsville
3020 University Dr., Hunlsville

Olensky Bros.
3763 Airport Blvd., Mobile

Arizona

Ham Shack
450 6-A N. 16th St., Phoenix

Millets TV & Radio
621 East Broadway, Mesa

California

Byte Shop
8038 Clairmont Mesa Blvd.,
San Diego

Byte Shop
123 E. Yorba Linda, Placentia

Byte Shop of Mt. View
1415 West El Camino Real, Ml. View

Byte Shop of Sacramento
6041 Greenback Ln., Cilrus Heights

Capital Computer Syslems
3396 El Camino Ave., Sacramento

Computers Made Easy
819 East Ave. Q-9, Palmdale

Computer Store of San Leandro
701 MacArthur Blvd., San Leandro

Computer World
6791 Westminster Ave., Westminster

Computerland
16720 S. Hawthorne, Lawndale

Compulerland of W. LA
6840 La Cienega Blvd., Inglewood

Coast Electronics
3118 No Main St., Morro Bay

Computerland
24001 via Fabricante No 904,
Mission Viejo

Hobbi-tronics
1378 So. Bascom Ave., San Jose

Hobby World
19511 Business Clr. Dr., Unit 6
Borthridge

I.C.E. House Inc.
398 North E. St., San Bernardino

Jade Compuler Products
4901 W. Rosecrans, Hawthorne

Marfam Co.
6351 Almaden Rd., San Jose

Opamp/Technical Books
1033 N. Sycamore Ave., Los Angeles

Q.I. Computers, Inc.
15818 Hawthorne Blvd., Lawndale

Radio Shack Dealer
8250 Mira Mesa Blvd., San Diego

Radio Shack Dealer
50 N. Cabrillo Hwy., Halt Moon Bay

Santa Rosa Computer Cenler
604 7th St., Santa Rosa

Silver Spur Elect. Comm.
13552 Central Ave., Chino

The Computer Slore
820 Broadway, Santa Monica

Colorado

Byte Shop
3464 S. Acoma St., Englewood

Colorado Computer Syslems
311 W. 74th Ave., Westminster

Computerland of North Denver
8749 Wadswoith Blvd., Arvada

Computer Shack
1635 South Prairie, Pueblo

The Computer Slore
2300 Welton St.. Denver

Connecticut

American Business Computers
454 Thames St., Groton

Computerlab
130 Jefferson, New London

Computerland
1700 Post Rd., Fairfleld

Computerland
60 Skiff St., Hamden

Computer Works
1439 Poet Rd. E., Liberty Plaza,
Westport

D.C.

The Program Store
4200 Wisconsin Ave., N.W.,
Washington, D.C.

Florida

Adventure International
200 Bald Cypress Ct., Longwood

AMF Electronics
11146 N. 30th St., Tampa

Boyd-Ebert Corporation
1328 West 15th St., Panama Cily

Compuler Center
6578 Central Ave., St. Petersburg

Computerland of Ft. Lauderdale
3963 N. Federal Hwy., Ft Lauderdale

Compulerland ot Jacksonville
2777-6 University Blvd. W.
Jacksonville

Compulerland of Tampa
1520 E. Fowler Ave., Tampa

Computer Shack
3336 Beach Blvd., Jacksonville

Curtis Waters Enterprises
236 Talbot Ave.. Melbourne

Heath Kit Electronic
4705 W 16th Ave. Center, Hialeah

HIS Compulermation
1295 Cypress Ave., Melbourne

Sound Ideas
2201-C N.W. 13lh, Gainesville

Ukatan Computer Store
Airport Rd., Destin

Williams Radio & TV Inc.
2062 Liberty St., Jacksonville

Georgia

Allanta Computer Mart
Allanta

Computerland of Allanta
2423 Cobb Parkway, Smyrna

Hawaii

Compulerland ot Hawaii
567 N. Federal Hwy., Honolulu

Radio Shack Assoc. Store
1712 S. King St., Honolulu

Idaho

Electronic Specialists
8411 Fairview Ave., Boise

Illinois

Computerland
4507 North Sterling, Peoria

Compulerland
9511 N. Milwaukee Ave., Niles

Computer Station
3659 Nameoki Rd., Granite City

Midwest Micro Computers, Inc.
708 S. Main Sl., Lombard

Kansas

Cenlral Kansas Computers
6 S. Broadway, Herington

Maine

Main Computronics
Intown Plaza, Bangor

Radio Shack
315 Main Mall Rd., So. Portland

Maryland

Jack Fives Electronics
4608 Debilen Circle, Pikesville

The Comm Cenler
9624 Ft. Meade Rd., Laurel

Massachusetts

ComputerCity
175 Main St., Charlestown

ComputerCity
50 Worcester Rd., Framingham

Computerland of Boston
214 Worcester Rd., Wellesley

Computer Packages Unllmited
244 W. Boylston St., West Boylston

Lighthouse Computer Software
14 Fall River Ave., Rehobath

New England Electronics Co.
679 Highland Ave., Needham

The Computer Slore
120 Cambridge St., Burlington

Tutts Radio & Electronics
206 Mystic Ave., Medford

Michigan

Computer Center
28251 Ford Rd., Garden City

Computer Connections
38437 Grand River, Farmington Hills

Computerland of Grand Rapids
2927 28th St. S.E., Kentwood

Computerland of Rochester
301 S. Livernois, Rochester

Computerland of Southfield
29673 Northwestern Hwy., Southfield

Computer Mart
560 W. 14 Mile Rd., Clawson

Hobby House
1035 W. Territorial Rd., Battle Creek

The Alternate Source
1806 Ada, Lansing

Ye Olde Teacher Shoppe
1823 Witmyre St., Ypsilanti

Minnesota

Computerland of Hopkins
11319 Hwy F., Hopkins

Digital Den
Burnsville Center

Minnesota Software Inc.
5422 Fisher St., White Bear Lake

Zim Computers
5717 Xerxes Ave., N. Brooklin Center

Mississippi

Dyer's, Inc.
200 E. Main St., West Point

Softwarehouse
816 Foley Sl., Jackson

W. Vernon Foster Inc.
816 Foley St., Jackson

Missouri

Computervan, Inc.
51 Florissant Oaks Shopping Cenler
Florissanl

Consolidated Software
16501 Greenwald Court, Belton

Montana

Intermountain Computer
529 So. 9th Sl., Livingston

Personal Computer
121 Red Oak Dr., Carl Junction

The Computer Store
1216 16th St. W. #35, Billings

Nebraska

Computerland of Omaha
11031 Elm St., Omaha

Midwest Computer Co. Inc.
8625 I St., Omaha

Midwest Computer Co. Inc.
4442 S. 84th St., Omaha

Midwesl Computer Co. Inc.
4403 S. 87th St., Omaha

Scottsbluff Typewriters Inc.
1824 Broadway, Scottsblutf

Nevada

Century 23
4566 Spring Mountain Rd., Las Vegas

New Hampshire

Bitsnbytes Computer Center
568 Pleasant St., Concord

ComputerCity
1525 S. Willow, Manchester

Paul's TV
Main St., Fremont

Portsmouth Computer Center
31 Raynes Ave., Portsmouth

Radio Shack Assoc. Store
Fairbanks Plaza, Keene

New Jersey

Computer Encounter
2 Nassau St., Princeton

Computerland
35 Plaza Rte. #4, W. Paramus

Computer Mart of NJ
501 Rte. 27, Iselln

Dave's Electronics
Pennsville Shopping Ctr., Pennsville

GHB Enterprlses Inc.
Rte. 38, Rudderaw Ave., Mapleshade

Personal Compuling Inc.
51 Centrat Sq., Llnwood

Radlo Shack/J&J Electronic
Mansfleld Shopping Ctr.
Rt. 57 Allen Rd., Hackettstown

The Bargain Brothers
Glen Roc Shopping Center
216 Scotch Road, Trenton

The Computer Emporium
Btdg. 103, Avenues ot Commerce
2428 Rte. 38, Cherry Hlll

New Mexlco

Autel Electronics Co.
232 Wisconein N.E., Albuquerque

Legey and Associates
2908 Tahiti Ct. N.E., Albuquerque

Mitchell's Muelc (Radio Shack)
407 W. Church, Carlsbad

South Weet Computer Center
121 Wyatt Drive, Sulte 7, Las Cruces

New York

Aristo Craft
314 Fifth Ave., NYC

Bits & Bytes
2800 Straight Rd., Fredonia

Computer Corner
200 Hamilton Ave., While Plains

Computer Era Corp.
1570 3rd Ave., New York

Compuler Factory
485 Lexington Ave., NYC

Compuler House, Inc.
721 Atlanlic Ave., Rochesler

Computerland of Nassau
79 Westbury Ave., Carle Place

Computer World
519 Boston Post Rd., Port Chester

Comtek Electronics, Inc.
2666 Coney Island Ave., Brooklyn

Comtek Electronics, Inc.
Staten Island Mall
Slore 220A, Staten Island

Digibyte Systems Corp.
31E. 31sl St., New York

Home Computer Center
671 Monroe Ave., Rochester

Key Electronics
Schenectady

Lashen Electronics Inc.
21 Broadway, Denville

Mr Computer
Imp Plaza, Rte. 9, Wappingers Falls

Sofiron Systems
306 Columbia Turnpike, Rensselaer

The Computer Tree Inc.
409 Hooper Rd., Endwell

Upstate Computer Shop
629 French Rd , Campus Plaza
New Hartford

North Carolina

Byle Shop ol Raleigh
1213 Hillsborough Sl., Raleigh

Ohio

Altair Business Systems, Inc.
5252 North Dixie Dr., Dayton

Astro Video Electronics
504 E. Main St., Lancaster

Cincinnati Computer Slore
4816 Interstate Dr., Cincinnati

Computerland
4579 Great Northern Blvd.,
N.Olmstead

Compulerland
6429 Busch Blvd., Columbus

Computerland
1288 Som Rd., Mayfield Heights

Compuler Store of Toledo
18 Hillwyck Dr., Toledo

Forbees Microsyslems Inc.
35 N. Broad, Fairborn

Microcomputer Center
7900 Paragon Rd., Dayton

Micro-Mini Compuler World
74 Robinwood, Columbus

Universal Amateur Radio, Inc.
1280 Aida Dr., Columbus

Oklahoma

Sounds, Elc.
Hyw. 33, Watonga

Vern Street Products
Radio Shack Dealer
114 W. Taft St., Sapulpa

Oregon

Computerland of Portland
12020 S.W. Main St., Tigard

Computer Pathways Unlimited, Inc
2151 Davcor St. S.E., Salem

TRS-80 Products Lld.
3520 S.E. Vineyard Rd., Portland

Pennsylvania

Artco Elect.
302 Wyoming Ave., Kingston

Artco Elect.
Back Mountain Shop. Ctr.
Shavertown

Audlo Mart
518 Fifth Ave., New Brighton

Computer Workshoppe
3848 William Penn Hwy, Monroeville

Computerland of Harrisburg
4644 Carlisle Pfke, Mechanicsburg

Erie Computer Co.
2127 West 8th St., Erie

Mlghty Byte Computer Center
537 Easton Rd., Horsham

Personal Computer Corp.
24-26 West Lancaster Ave., Paoli

Personal Computer Corp.
Frazer Mall, Lancaster Ave., Fiazer

Rhode Island

Computer City
165 Angell St., Providence

South Dakota

CB Radio Shack
21st and Broadway, Yankton

Tennessee

Compulerlab
671 S. Menden Hall Rd , Memphis

H & H Electronics Inc.
509 N. Jackson St., Tullahoma

Texas

Computer Port
926 N. Collins, Arlington

Houston Computer Tech
5313 Bissonet, Bellarie

Interactive Computer
7620 Dashwood, Houston

K.A. Elect.
9090 Stemmons Frwy., Dallas

Pan American Elect. Inc.
t117 Conway, Mission

Radio Shack Dealer
21969 Katy Freeway, Katy

Ram Micro Systems
6353 Camp Bowie Blvd., Ft. Worth

Waghalter Books Inc.
3 Greenway Plaza E., Houston

Utah

DC Computer Co.
1911 West 70 South, Provo

Qualily Technology
470 E. 2nd So., Sall Lake City

Virginla

Computer Works
Rte. 6, Box 65A, Harrisonburg

Home Computer Center
2927 Virginia Beach Blvd
Virginia Beach

Soulhside Radio Comm.
135 Pickwick Ave., Colonial Heignls

Washington

American Mercantile Co Inc.
2418 1st Ave. S., Seattle

Byte Shop ol Bellevue
14701 N.E. 20th Sl., Bellevue

Computerland of South King Co.
1500 S. 336 St., Suile 12
Federal Way

Magnolia Micro Systems
2812 Thorndyke Ave., Seattle

Personal Computers
S 104 Freva, Spokane

Ye Old Computer Shop
1301 G. Washington, Richland

West Virglnia

The Computer Corner Inc.
22 Beechurst Ave., Morgantown

The Computer Slore
Municipal Parking Bldg., Charleston

Wlsconsin

Byte Shop Of Milwaukee
6019 West Layton Ave., Greenfield

Computerland
690 S. Whitney Way, Madison

Petted Microsystems
4265 W. Loomis Rd., Milwaukee

Wyoming

Computer Concepts
1104 Logan Ave., Cheyenne

Puerto Rico

The Microcomputer Store
1568 Ave. Jesus T. Pinero
Caparra Terrace

Canada

CANADIAN DISTRIBUTORS.
Micron Distributing
409 Queen St., W. Toronto, Ont.
M5V 2A5

Computerland of Winnipeg
715 Portage Ave., Winnipeg, Man.

Compumart
411 Roosevelt Ave., Ollawa, Ontario

Micromalic Systems Inc.
101 8136 Park Rd., Richmond, B.C.

Micro Shack ot W Canada
333 Park Street, Regina. Sask.

Orthon Holdings Ltd.
12411 Stony Plain Road
Edmonton, Alberta

Tolal Computer Systems
Ajax, Ontario

Austria

Byte Shop Microelectronic
Favoritenstr. 20, Wien

England

Mighty Micro
33 Cardiff Rd., Watford Herts

France

Sivea s.a.
20, Rue de Leningrad, Paris

Italy

HQMIC s.r.l.
Piazza De Angeli 1, Milano

Switzerland

Tandy Corp.
Basterstr. 145, Zuerich

West Germany

EUROPEAN DISTRIBUTOR:
Microshop Bodensee
Markstr. 3, 7778 Markdorf

AAA-Electronic
Habsburgerstr. 134, Freiburg

Basic Sottware und Schulung
Constantinstr. 88, Koeln 21

Electronic Hobby Shop
Maximilianstr 22, Bonn

Henniger Computers
Landwehrstr. 39, Muenchen 2

Microcomputer Center
Alsfelderstr. 7, Darmstadt

Muenzenlohr
Toelzerstr. 5, Holzkirchen

R + R Elec!ronic
Adlerstr. 55, Heidelberg

Australia

Computer Country Pty Ltd
5 Tonkin Ave., Balwyn VIC

Deforest Sottware
36 Glen Tower Drive
Glen Waverly, VIC

Sure-Load Software
P.O. Box 26, Weston, A.C.T

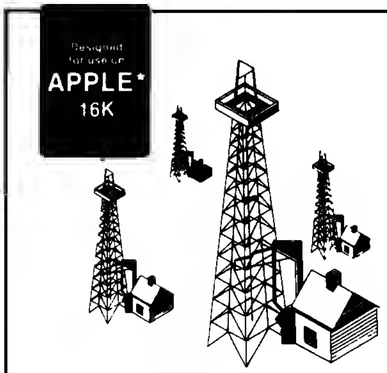
South Africa

SOUTH AFRICAN DISTRIBUTOR:
Eddie Talberg
P Q Box 745, Johannesburg

Instant Software™ Inc.

Peterborough, N.H.
03458

Instant Software™ New Releases



*A trademark of Apple Computer Inc

Oil Tycoon

You intend to become the most powerful person in the petroleum industry. Your competitor is seeking the very same goal. There's room for only one of you at the top!

Begin with two million dollars, invest in Research and Development to lower exploration costs and reduce the likelihood of a dry well. After exploring a well site, you must decide whether to drill or to look further. Once you have oil to sell, you can set your own price. But if you get *too* greedy, demand will drop dramatically!

When this two person game is over, you may find yourself the wealthiest tycoon on the block—or you may be the victim of too many oil spills, blowouts and tanker disasters. However you fare in the end, you'll have fun building your empire with Oil Tycoon.

Apple II 16K & Applesoft in ROM. **Order no. 0079A $9.95**

Santa Paravia and Fiumaccio

NOW AVAILABLE FOR APPLE AND PET

It is the dawn of the 15th Century; you rule an tiny Italian city-state. Your goal: The Crown!

Up to six players can compete as rulers of neighboring cities. You control the grain harvest, feed your serfs, set tax rates dispense justice and invest in public works.

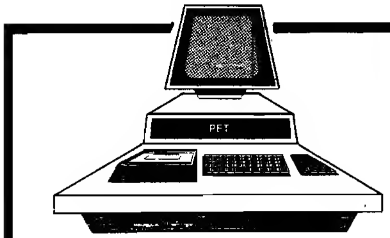
The future of your realm will depend on your decisions. If they are wise, your city-state will grow and you will acquire loftier titles. If your rule is incompetent, your people will starve and you may be invaded by your neighbors.

How will *you* rule your kingdom? Will you be an enlightened leader—or an unscupulous despot? Only you can answer that question—with Santa Paravia and Fiumaccio from Instant Software.

Super graphics versions now available for the Apple II.

Apple II 48K & Applesoft in ROM. **Order no. 0174A $9.95**

PET 16K. **Order no. 0175P $9.95**



PET UTILITY I

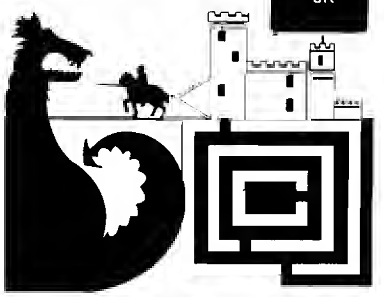
You're working under a serious handicap if you can't write programs in machine language. The PET Utility I package gives you the tools you need:

- **•Monitr**—The Monitr program lets you write, edit, save, and verify any machine language and/or BASIC program. Just load and run the Monitr program and then load the program you want to edit.
- **•Programmer's Calculator**—This program will convert numbers into the binary, octal, decimal, and hexadecimal systems and function as a floating point calculator. It will also display all four numbering systems simultaneously and allow you to handle large numbers limited only by the size of your screen.

For the 8K PET. **Order No. 0105P. $9.95.**

Chimera

Designed for use on PET* 8K



* A trademark of Commodore Business Machines Inc.

If you think the legendary Chimera was hard to handle, wait until you try this package. Included are:

- **•Dropoffs**—You must make your opponent's men "dropoff" the board by moving and firing your own men. For one or two players.
- **•Dots**—Place your lines carefully as you try to build and capture the squares. For one player.
- **•Batter-up**—You and another player take turns at bat as your PET becomes both the pitcher and the umpire. For two players.
- **•Reflex**—Round and round the little white ball rolls. Only fast reflexes can guide it into the center of the maze.

You'll almost be able to feel the Chimera's fiery breath as you play the games on your 8K PET. **Order No. 0110P. $9.95.**

Look for Instant Software at a store near you. See dealer list on opposite page. If the store nearest you does not stock Instant Software, use this order blank to purchase your software directly or call Toll-Free 1-800-258-5473.

Name ____________________

Address ____________________

City ____________________

State ____________ Zip ____________

□ Check □ Money order

□ VISA □ AMEX □ Master Charge

Card No. ____________________

Expiration Date ____________________

Signed ____________ Date ____________

Order your Instant Software today!

Quantity	Order No.	Unit Cost	Total Cost
		Handling	$1.00
		Total Order	

Instant Software Inc. Dept. 680FO
Peterborough, N.H. 03458 USA

Prices subject to change without notice.

Instant Software™ Inc.

PETERBOROUGH, N.H. 03458
603-924-7296

OHIO SCIENTIFIC'S

Welcome to the Ohio Scientific Small Systems Journal. This is the first issue of a continuing monthly feature in Micro.

The Small Systems Journal will, hopefully, serve two major purposes. The first is to expand and enhance the knowledge of Ohio Scientifc's systems for those of you who already own or use an Ohio Scientific computer. The second is to introduce non-OSI users to some of the systems.

We will try to maintain a balance in the Journal of technical and nontechnical articles, with both hardware and software features. This issue covers three topics:

Simple Modem Routine for C4P MF and C8P DF
User Modifiable I/O for the C1P
The UTI — Universal Telephone Interface

In future issues we hope to cover the topics of interest to the most people. To this end we invite suggestions, on article content, to be submitted to:

Ohio Scientific, Inc.
Small Systems Journal
1333 S. Chillicothe Road
Aurora, Ohio 44202

CUSTOMIZING C1P BASIC-in-ROM I/O

One little known feature of Ohio Scientific's Challenger 1P computer is that the input and output (I/O) routines for ROM BASIC are user modifiable. This is made possible by the jump (JMP) indirect feature of the 6502 processor.

For example, when BASIC requires a character to be input, it executes the following subroutine call:

```
XXX 20EBFF        JSR BASIN
```

The actual input routine is called via indirection of the INVCT (in vector) register pair (hex 0218 and 0219).

This means that the processor instead of executing a "normal" jump (Ie., JMP destination) fetches the data contained in the indirct register pair (218 and 219 in this case) and uses those 16 data bits as the address of the jump's destination. The net effect is that BASIC's input call is handled by a machine code subroutine whose address is stored at (hex) locations 218 and 219.

All of BASIC's input call is handled in this fashion. The following table gives the address of each routine and the related register pair:

TABLE 1

Routine	Address	Register-Pair	Function
BASIN	FFEB	218, 219	Get character input in accumulator
BASOUT	FFEE	21A, 21B	Output character in accumulator
CNTRLC	FFF1	21C, 21D	Check CONTROL-C in accumulator
BLOAD	FFF4	21E, 21F	BASIC LOAD keyword
BSAVE	FFF7	220,221	BASIC SAVE keyword

By studying Table 1, it may be seen that although the C1P BASIC is in ROM and that BASIC I/O calls are in ROM, the actual addresses of the I/O routines are in RAM.

When the C1P is reset, the five indirect register pairs are initialized with routine addresses contained in the BASIC support ROM. However,after starting BASIC, these addresses may be changed to point to custom routines with appropriate POKE's.

The following is an example of one of many possible user created routines. This one offers a modification to BASIC's input routine.

Normally, when an input character is deleted (with shift-0), BASIC responds by removing the character from it's imput buffer and reoutputting the cursor. The routine in Listing 1 removes the character from both the buffer and the CRT display.

LISTING 1

```
 10                ;
 20                ;SIMPLE 1P CHAR DELETE
 30                ;
 40                ;
 50 0222                  *=$222
 60                ;
 70 0222 2000FD KEYIN  JSR $FD00    GET KEY
 80 0225 C95F          CMP #$5F     SHIFT 0?
 90 0227 F001          BEQ *+3      YES
100 0229 60            RTS          NO. SEND CHAR TO BASIC
110                ;
120 022A 8A            TXA          SAVE X
130 022B 48            PHA
140 022C A920          LDA #$20     SPACE IN A
150 022E AE0002        LDX $200     GET CRT INDEX
160 0231 20CDBF        JSR $BECD    REMOVE CURSOR
170 0234 CA            DEX          ADJ INDEX
180 0235 20CDBF        JSR $BFCD    REMOVE LAST CHAR
190 0238 8E0002        STX $200     SAVE NEW CRT INDEX
200 023B 68            PLA
210 023C AA            TAX          RESTORE X
220 023D CA            DEX          ADJ BASIC'S BUFFER INDEX
230 023E 4C2202        JMP $222     GET ANOTHER CHAR
240                ;
250                       .END
```

The first listing is assembled at (hex) 222 and occupies a small portion of the free RAM space from 222 through 2FA.

The overall operation of the routine is as follows:

1) A character is read from the keyboard and if it's not shift-0, simply return to BASIC.

2) If shift-0 is received, both the current cursor position and the previous character position are removed from the screen.

3) BASIC's buffer index is decremented and another character is input.

Note that within the routine, the original contents of th X-register (BASIC's buffer index) is saved. Generall speaking, the original contents of all processor register (except the accumulator in BASIN and CNTRLC c course) should be protected when interfacing to BASIC

Listing 2 demonstrates a typical method for overlaying the code via BASIC.

LISTING 2

```
 10 REM
 20 REM SIMPLE 1P CHARACTER DELETE
 30 REM
 40 REM 6502 CODE
 50 DATA 32,0,253
 55 DATA 201,95,240,1,96
 60 DATA 138,72,169,32
 65 DATA 174,0,2
 70 DATA 32,205,191
 75 DATA 202,32,205,191
 80 DATA 142,0,2
 85 DATA 104,170
 90 DATA 202,76,34,2
100 REM
110 REM SET-UP 6502 CODE
120 FOR I=0 TO 30
130 READ D
140 POKE 546+I,D
150 NEXT I
200 REM
210 REM OVERLAY NEW INPUT VECTOR
220 POKE 536,34
230 POKE 537,2
999 END
```

The DATA statements in lines 50 through 90 contain the decimal equivalent of the hexidecimal numbers in the assembly code. That is, the (hex) bytes 20,00,FD equal the decimal 32,0,253; etc. Lines 120 through 150 POKE the code into thirty-one consecutive locations starting at decimal 546 (hex 0222).

The final statements, line 220 and 230, enable the use of the routine by changing the address at the input indirect register pair 536, 537, (hex 218, 219). Note that the address POKEd into this register pair is low, high order. This means that (hex) address 0222 is actually represented in memory thusly:

0218 contains 22
0219 contains 02

This format is typical for all 6502 indirect addressing schemes.

After the routine has been loaded, it should never be necessary to reload it. This of course assumes power is not interrupted and the routine is not over written. Remember, however, that the original input vector is rewritten on reset. To reimplement this routine after reset, only BASIC lines 220 and 230 need be re-executed.

Note, if done in the immediate mode, be sure to execute both POKE's on one line:

POKE 536,34: POKE 537,2

If not, the input vector will not point to either routine and an error will surely occur.

As this routine is intended primarily as a demonstration of an interfacing method, not a field upgrade, there are a couple of limitations that could be overcome by additional code:

1) The cassette I/O is no longer polled on input. this means that LOAD function is essentially disabled. This is easily overcome by a reset and Warm Start.
2) No check of BASIC's buffer index is done. It's possible to delete past the beginning of a line. To delete and entire line, shift-P should be used.

A hint to fix problem #2; never let 'X' (line 220 of the assembly code) go below zero.

By following this general example, you should be able to implement several interesting I/O routines. One suggestion might be to try a memory I/O in parallel with the normal I/O. Remember, in addition to the 216 bytes (decimal) available from (hex) 222 through 2 FA, areas of "upper" memory may also be reserved when responding to MEMORY SIZE? at Cold Start.

Modem Routine for C4P MF and C8P DF

With non-prime time rates becoming available on large time sharing systems, a modem is rapidly becoming a very useful addition to a home computer system.

The following program is designed for use with a standard modem (with RS-232) and and Ohio Scientific C4P MF or C8P DF computer.

The routine is essentially a "bare-bones" system which allows your computer to be used as a terminal for the timeshare system accessed. This program may certainly be expanded to whatever terminal intelligence you may require.

List of Control Commands

CONTROL-S — Stop print
CONTROL-Q — Re-start print (use with CNTRL-S)
CONTROL-D — Toggle duplex mode between Full and Half
CONTROL-E — Toggle Auto-Echo (enable/disable)
CONTROL-B — Return to subroutine caller

The program is a subroutine that may be called by BASIC via the USR function. Note, in this case, CONTROL-B will cause a return to BASIC.

This program is assembled to operate under OS-65D V3.2 Home Control Operating System.

MODEM ROUTINE FOR C4P MF AND C8P DF

```
 10 4000               *=$4000
 20
 30 2599=              CRTOUT=$2599
 40 2AC5=              DEFAUL=$2AC5
 50 3180=              KBDIN=$3180
 60 2644=              KBSWAP=$2644
 70 267A=              MRKT=$267A
 80 2343=              OUTCH=$2343
 90 2322=              OUTFLG=$2322
100 00E5=              SFLAG=$00E5
110 24CD=              TTYOUT=$24CD
120
121 4000 A934          LDA $52       SET MODEM PORT
122 4002 8D03F7        STA 63235     THATS DECIMAL FOLKS
123 4005 A902          LDA $02       SELECT 300 BAUD
124 4007 8D00FC        STA $FC00
```

OHIO SCIENTIFIC'S

```
130 400A AD2223        LDA OUTFLG   GET SELECTED DEVICE
140 400D 202C41        JSR PRINTR   SET UP OUTPUT FOR DEVICES
                                    SELECTED
150                ;
160 4010 ADC62A        LDA DEFAUL+1 GET DEFAULT DEVICE
170 4013 C902          CMP $2       IS IT VIDEO?
180 4015 F014          BEQ START    YES, CODE IS OKAY START
                                    EXECUTION
190                ;
200 4017 A940          LDA #KBSERL/256
210 4019 8D4840        STA KEY+2    SET UP I/O FOR A
220 401C A9CB          LDA #KBSERL  SERIAL TERMINAL
230 401E 8D4940        STA KEY+1
240 4021 A924          LDA #TTYOUT/256
250 4823 8D5A40        STA KBD1+2
260 4826 A9CD          LDA #TTYOUT
270 4028 8D5940        STA KBD1+1
280                ;
290 402B 204426 START  JSR KBSWAP   DO A SWAP IN CASE KBD POLLED
300                ;
310                ;
315 402E=          P1=*
320 402E AD00FC MOREAD LDA $FC00    READ THE MODEM?
330 4031 4A            LSR A        IS THERE A BYTE READY?
340 4032 9011          BCC KDYBRD   NO. CHECK KBRD FOR INPUT
350 4034 AD01FC MODMIN LDA $FC01    YES, GET THE BYTE
360 4037 ACCA40        LDY AUTOEC   CHK AUTO ECHO ?
370 403A F003          8EQ OUTCHR   NO. OUTPUT CHAR
380 403C 206240        JSR MOWRIT   YES, ECHO IT TO THE MODEM
390 403F 20F940 OUTCHR JSR CHECK    CHK IF STOP PRINT NEEDED OVER
                                    MODEM
400 4042 204323        JSR OUTCH    OUTPUT CHAR
410                ;
420 4045 20D740 KEYBRD JSR CHECKS   CHK IF START PRINT NEEDED OVER
                                    MODEM
430 4048 208031 KEY    JSR KBDIN    GO GET ANY KEYS DEPRESSED
440 404B F0E1          BEQ P1       NO KEYS. START LOOP OVER
450 404D 206F40        JSR KEYCHK   CHK FOR ANY SPECIAL KEYS
460 4050 F0DC          BEQ P1       YES, ONE EXECUTED
470 4052 ACC940 KBD    LDY DUPLEX   DUPLEX FLAG SET
480 4055 F005          BEQ KBD2     NO. WRITE CHAR TO MODEM
490 4057 48            PHA          YES
500 4058 209925 KBD1   JSR CRTOUT   OUTPUT CHR AT LOCAL TERMINAL
                                    FIRST
510 4058 68            PLA          RESTORE A
520 405C 206240 KBD2   JSR MOWRIT   WRITE KEY DEPRESSION TO MODEM
530 405F 4C2E40        JMP P1       START LOOP OVER
540                ;
550 4062 48     MOWRIT PHA          SAVE A
560 4063 AD00FC        LDA $FC00    WAIT FOR READY TO SEND
570 4066 4A            LSR A
580 4067 4A            LSR A        READY?
590 4068 90F9          BCC MOWRIT+1 NO. CHK AGAIN
600 406A 68            PLA          YES, RESTORE A
610 406B 8D01FC        STA $FC01    SEND 8YTE
620 406E 60     AHRTS  RTS          ALL DONE
630                :
640 406F C902   KEYCHK CMP #'8-$40  CTRL-B ? (HANG UP PHONE)
650 4071 F01F          BEQ GOODBY   YES, HANG UP AND RETURN
660 4073 C904          CMP #'D-$40  CTRL-D ? (DUPLEX E/D)
670 4075 F047          8EQ FLPDUP   YES, CHANGE FLAG
680 4077 C905          CMP #'D-$40  CTRL-E ? (AUTO ECHO E/D)
690 4079 F038          8EQ FLPECO   YES, CHANGE FLAG
700 4079 C919          CMP #'Y-$40  CTRL-Y ? (SERIAL PRINTER E/D)
750 4083 AD2223 PARALL LDA OUTFLG   YES, SEE WHAT DEVICES WE HAVE
760 4086 C902          CMP #2       VIDEO ?
770 4088 F0E4          BEQ AHRTS    YES, DON'T TURN ON PARL PNTR
780 408A 4908          EOR #8       NO, CHANGE OUTPUT FLAG
790 408C 202C41        JSR PRINTR   GO CHANGE CODE TO SUPPORT
                                    CONDITIONS
800 408F A900          LDA #0       CLEAR A
810 4091 60            RTS          ALL DONE
820                ;
830 4092 68     GOOD8Y PLA          RETURN TO THE RIGHT PLACE
840 4093 68            PLA
850 4094 ADC62A        LDA DEFAUL+1 SET BACK TO DEVICES
860 4097 8D2223        STA OUTFLG   WE STARTED WITH
870 409A ADF240        LDA CTRLQ+1  SEND START PRINT JUST IN CASE
880 409D 206240        JSR MOWRIT
890 40A0 204426        JSR KBSWAP   RETURN POLLED KEYBOARD
900 40A3 60            RTS          RETURN TO CALLER
```

```
 910               ;
 920 40A4 AD2223 SERIAL LDA OUTFLG   SEE WHAT DEVICES WE HAVE
 930 40A7 C901          CMP #1       SERIAL ?
 940 40A9 F0C3          BEQ AHRTS    YES, DON'T TURN ON SERIAL PNTR
 950 40AB 4901          EOR #1       NO, CHANGE OUTPUT FLAG
 960 40AD 202C41        JSR PRINTR   GO CHANGE CODE TO SUPPORT
                                     CONDITIONS
 970 40B0 A900          LDA #0       CLEAR A
 980 40B2 60            RTS          ALL DONE
 990               ;
1000 40B3 ADCA40 FLPECO LDA AUTOEC   CHANGE AUTO ECHO FLAG
1010 40B6 4901          EOR #1       ENABLE/DISABLE
1020 40B8 8DCA40        STA AUTOEC
1030 40BB A900          LDA #0
1040 40BD 60            RTS
1050               ;
1060 40BE ADC940 FLPDUP LDA DUPLEX   CHANGE DUPLEX FLAG
1070 40C1 4901          EOR #1       ENABLE/DISABLE
1080 40C3 8DC940        STA DUPLEX
1090 40C6 A900          LDA #0
1100 40C8 60            RTS
1110               ;
1120 40C9 00     DUPLEX .BYTE 0      DUPLEX FLAG
1130 40CA 00     AUTOEC .BYTE 0      AUTO ECHO FLAG
1140               ;
1150 40CB AD00FC KBSERL LDA $FC00    SERIAL KEYBOARD INPUT ROUTINE
1160 40CE 4A            LSR A
1170 40CF A900          LDA #0
1180 40D1 9003          8CC KBSER1
1190 40D3 AD01FC        LDA $FC01
1200 40D6 60     KBSER1 RTS
1210               ;
1220 40D7 A5E5   CHECKS LDA SFLAG    LOAD STOP PRINT FLAG
1230 40D9 CD0341        CMP CTRLS+2  HAS A STOP PRINT BEEN SENT ?
1240 40DC D01A          BNE OUT2     NO, RETURN
1250 40DE AE4841        LDX COUNT    YES, IS THERE ANY CHR IN BUFFER
1260 40E1 F00E          BEQ CTRLQ    NO, SEND START PRINT CODE
1270 40E3 A000          LDY #0       YES, GET CHR OUT OF 8UFFER
1280 40E5 894C41 LOOPER LDA 8UFFER,Y
1290 40E8 204323        JSR OUTCH    SHOW WHAT WE HAVE
1300 40EB C8            INY
1310 40EC CC4841        CPY COUNT    IS THAT ALL ?
1320 40EF D0F4          BNE LOOPER   NO, GET NEXT ONE
1330 40F1 A911   CTRLQ  LDA #'Q-$40  YES, SEND START PRINT COMMAND
1340 40F3 85E5          STA SFLAG    RESET FLAG
1350 40F5 206240        JSR MOWRIT
1360 40F8 60     OUT2   RTS          ALL DONE
1370               ;
1380 40F9 C90D   CHECK  CMP #$0D     IS CHAR A CRT OR LF ?
1390 40FB F004          8EQ CTRLS    IF YES, SEND STOP PRINT
1400 40FD C90A          CMP #$0A     IF NOT, RETURN
1410 40FF D02A          BNE OUT
1420 4101 48     CTRLS  PHA          SAVE A
1430 4102 A913          LDA #'S-$40
1440 4104 85E5          STA SFLAG    SET FLAG
1450 4106 206240        JSR MOWRIT   SEND STOP PRINT CODE
1460 4109 A200          LDX #0
1470 4108 8E4841        STX COUNT    SET # IN BUFFER TO 0
1480 410E AC7B26 LOOPY  LDY MRKT+1   WAIT 4 CHR TIME FOR ANY CHAR
1490 4111 AD00FC 8EGIN  LDA $FC00    STILL COMING
1500 4114 4A            LSR A        IS THERE A CHR WAITING ?
1510 4115 9007          BCC LOOP49   NO, CONTINUE WAIT
1520 4117 AD01FC        LDA $FC01    YES, SAVE CHARACTER
1530 411A 9D4C41        STA BUFFER,X
1540 411D E8            INX          NEXT ?
1550 411E 18     LOOP49 CLC
1560 411F 88            DEY
1570 4120 D0EF          8NE 8EGIN    LOOP AGAIN
1580 4122 EE4841        INC COUNT    DONE ?
1590 4125 D0E7          BNE LOOPY    NO, TRY AGAIN
1600 4127 8E4841        STX COUNT    SAVE # OF CHRS IN BUFFER
1610 412A 68            PLA          RESTORE A
1620 4128 60     OUT    RTS
1630               ;
1640 412C 8D2223 PRINTR STA OUTFLG   SAVE DEVICES SELECTED
1650 412F C909          CMP #9       IS EITHER PRINTER TURNED ON ?
1660 4131 F00D          BEQ PRTENA   YES, ENABLE PRINT START & STOP
1670 4133 C903          CMP #3
1680 4135 F009          8EQ PRTENA   YES, ENABLE PRINT START & STOP
1690 4137 A960   PRTDIS LDA #$60     NO, DISABLE PRINT START & STOP
1700 4139 8DD740        STA CHECKS
```

CALL 1-800-321-6850 TOLL FREE

```
1710 413C 8DF940         STA CHECK
1720 413F 60             RTS          RETURN
1730 4140 A9A5   PRTENA  LDA #$A5
1740 4142 8DD740         STA CHECKS
1750 4145 A9C9           LDA #$C9
1760 4147 8DF940         STA CHECK
1770 414A 60             RTS          ALL DONE
1780             ;
1790             ;
1800 414B 00     COUNT   .BYTE 0      COUNTER BYTE
1810             ;
1820 414C=               BUFFER=*     BUFFER FOR INCOMING CHRS
```

CA-15 Universal Telephone Interface

The Universal Telephone Interface (UTI) provides the broadest range of computer/telephone utilization options ever offered in a single product. The UTI occupies one slot of a C8P, C2-OEM or C3 series computer and connects directly to a normal telephone line via a FCC approved isolation module called a CBT. CBT's are available from many telephone companies on a monthly charge basis. However,Ohio Scientific also offers CBT's as a accessory item for user connection to telephone lines. The UTI can be connected in conjunction with one or more telephones on the line and can also operate as the only device on the line. No user intervention is required to initiate or answer calls. (i.E., total computer control is possible) The UTI compatable with Touch Tone or Pulse Dial (rotary dial) lines.

The UTI includes a conputer to telephone interface which allows the user (through software) to "connect" (pick up) and "disconnect" (hang up) the phone. The computer can detect a dial tone before dialing. Note, again this may be either pulse or tone dialing. After dialing a phone number,it can detect a ring on incoming calls. Additionally, the UTI includes a failsafe circuit which automatically hangs up the phone after 90 seconds of inactivity. This prevents accidental extended connection to the phone line.

The UTI allows any one of five different audio signals to be output to the phone line at any one time. These five signals are touch tones, taped messages, audio from an auxiliary device, modem signals and Votrax generated synthetic speech (when equipped with the Votrax option). Touch tones can be output to the phone at any time. This allows the computer to tone dial the phone, or to signal humans or other computers.

The UTI can output taped messages to the phone from cassette tape recorders. The tape recorder can be automatically turned off and on with a motor control signal which is provided by the UTI.

Audio from a digital-to-analog converter,radio or other auxiliary device can be selected for output to the phone line through the auxiliary jack on the UTI.

The "on board" modem outputs 300 baud data to the phone. The modem features "originate" or "answer" modes.

Votrax equipped UTI's can output Votrax generated synthetic speech to the phone. The UTI contains an "on board" amplifier which will drive an 8 ohm speaker with Votrax speech. This allows the Votrax to also be operated in a "stand alone" mode.

Input from the phone can include voice, touch tones, and modem signals. Voices can be routed to an audio amplifier via the UTI auxiliary audio output connector and/or to an auxiliary cassette tape recorder which can be automatically turned off and on with a motor control signal which is provided by the UTI. Touch tones can be decoded by the UTI. This allows remote control of your computer via touch tones. Modem signals are routed to the "on board" 300 baud modem.

The addition of a UTI to your OSI computer will allow computer to computer,computer to human, and human to computer communications. Several typical applications of these types of communication are presented below.

The UTI will allow computer to computer communication by operating as a conventional auto-answer, auto-dial, answer or originate 300 baud modem with "hands off" operation. This allows OSI computers to be remotely timeshared economically as well as being the basis of computer bulletin board services. Another application is the automatic interrogation of two UTI based computers for remote process control or remote data acquisition.

An OSI computer,when equipped with a UTI and OSI's security monitor, can provide computer to human communication by providing a complete security system including the monitoring if a home, business, warehouse, vacation home, boat, etc. This computer based security system can notify any telephone number (or numbers) with a Votrax or tape recorded message.

Applications of human to computer communication using a UTI based computer include allowing a caller's telephone to act as a "computer terminal", or allowing caller interrogation of security or system status using touch tone commands. When a UTI based computer is used in conjunction with the A.C. remote control and/or parallel I/O, it allows a caller to control home lights and appliances as well as control of automated processes. Of course, the program would allow this type of control to be used only by caller who has entered a correct password, thus giving only authorized persons control of these devices.

A UTI based computer equipped with a cassette tape recorder can act as a telephone answering device. In addition, if a remote rewind tape deck, endless tape or A/D-D/A system is added, the computer system can relay voice messages including caller initiated playback of recorded messages and caller stored messages for relay to another phone number at another time and/or to be attempted until the message gets through.

The UTI comes complete with documentation on how to use each feature as well as two copies of the OS-65D V3.2 "PHONE DEMO" diskettes.

Included are such programs as:

A) A "Home Monitor" demo which allows a touch tone equipped caller to interrogate the status of an AC-17P home security system. The caller may also send commands to lamps and appliances via the AC-12P A.C. Remote Control system.

B) An "Automatic Dial a Modem" program allows a UTI equipped computer to function as a timeshare terminal with "hands off" modem operation.

C) "Remote Computer Interrogation" demo program which requires two UTI computers. One computer automatically interrogates the other one. This demonstrates the capability of remote monitoring and process control systems.

Licensed Level 3 users will be provided with a version of a Level 3 system which will allow one partition of their timeshare system to be available for telephone communications.

Summary

The Ohio Scientific Universal Telephone Interface is a cost effective way to complete the link between the computer and the telephone.It allows the computer to easily access outside information via large time share systems and data bases. In addition, it allows remote interrogation and operation of the computer from any telephone. These are just a few of the more common applications of the Universal Telephone Interface, with unlimited applications to meet specific requirements.

® Votrax is a registered trademark of Federal Screw Works

OSI CHALLENGER SOFTWARE

Video Games — Board Games

Utilities — D&D Games

And More!

For all BASIC-in-ROM.
Some programs also on disk.

Send For Free Catalog
Dealer Inquiries Invited

Orion Software
Associates, Inc.
147 Main Street
Ossining, NY 10562

OHIO SCIENTIFIC

Hardware..C1P Video—gives true 32 or 64 chrs/line with guard bands. This is not a make-shift mod. It makes your video every bit as good as the 4P's plus you have switch selectable 1,2 and 3 MHz. CPU clock as well as 300, 600 and 1200 baud for cassette and serial port all crystal controlled.

Complete plans—$18.95, Kit $39.95 or send in your C1P to Personal & Business Computer Connection, 38437 Grand River, Farmington Hills, Mich 48018, and we will install the Video mod for $79.95. Other mods available..add sound; RS-232 port cassette motor control.

Software (with documentation) For C1, C2, 4P & 8P Chess 1.9, Backgammon, excellent card games, arcade type games, utility programs, mini word processor memory maps, etc.

Catalog with free program (hard copy) & memory map for BASIC in ROM models...$1.00

Progressive Computing
3336 Avondale Crt.
Windsor, Ontario
CANADA N9E 1X6
(519)969-2500

Price for kit in last month's issue was in error.

OSI SOFTWARE OSI
FOR
OHIO SCIENTIFIC

Over 50 programs for C1, C2, C4, & Superboard, on tape and disk. All come with listings and complete documentation.

GAMES-4K-Tape	
CHESS FOR OSI-specify system	$19.95
STARFIGHTER	5.95
Real time space war.	
SEAWOLFE	5.95
Floating mines, three target ships, etc.	
LUNAR LANDER	5.95
With full graphics.	
ALIEN INVADERS	6.95
Rows of menacing, munching monsters march on earth.	
8K GAMES	
BACKGAMMON	$9.95
BLACKJACK	6.95
Plays *all* Vegas rules.	
Add $1.00 each for Color/Sound.	

UTILITIES	
C1P TEXT EDITOR	$9.95
Gives real backspace, one key screen clear, *and midline editing.*	
RENUMBERER	5.95
SUPERUTILITY	12.95
Has renumberer, Variable table maker and Search.	
DISKS	
SUPERDISK	$24.95
(for C4MF or C8MF) contains text editor, renumberer, new VEXEC*	
THE AARDVARK/JOURNAL FOR OSI	
Six issues of user information.	$9.00

Our $1.00 catalog has free game and utility listings, programming hints and a lot of PEEKs and POKEs and other stuff that OSI forgot to mention—and a *lot* more programs for sale.

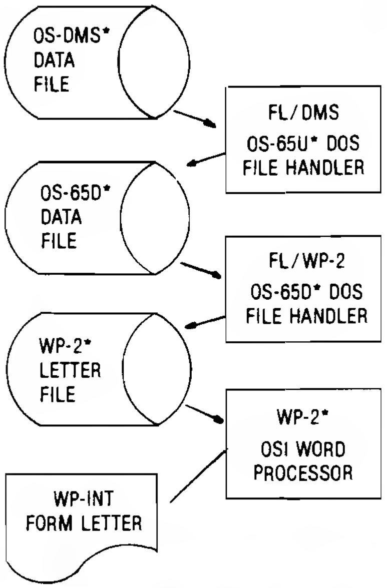
DISKS 5" COLOR/SOUND $29.95
DISK-1. STARFIGHTER, ROBOTANK, SEA WOLFE, BOMBER, TEN TANK BLITZ,
DISK 2 BREAKTHROUGH, LUNAR LANDER, ALIEN INVADER, KILLERROBOTS, SLASHBALL.

AARDVARK TECHNICAL SERVICES 91

1690 BOLTON, WALLED LAKE MI 48088 313-624-6316

OHIO SCIENTIFIC USERS

AT LAST! FORM LETTER GENERATION USING OS-DMS* and WP-2*



WP-INT	
disk & manual	$80.00
manual alone	$ 2.00

ASK YOUR DEALER FOR A DEMONSTRATION

D C S software products
PEOPLE ORIENTED COMPUTER SYSTEMS

* Requires OS-DMS and WP-2.

* OS-DMS, WP-2, OS-65U, and OS-65D ARE ALL PRODUCTS OF OHIO SCIENTIFIC.

TO ORDER WP—INT DIRECT
CHECK • MONEY ORDER • C O.D.
DCS SOFTWARE PROOUCTS
2729 Lowery Ct
Zion, IL 60099
Please add $1 50 on disk orders for shipping

BASIC and Machine Language with the Micromodem II

George J. Dombrowski, Jr.

There is no doubt that the Micromodem II produced by D.C. Hayes Associates for use with the Apple II computer is a very sophisticated telecommunications device. I purchased a Micromodem several months ago and have been pleased with its performance ever since. This device couples directly with Ma Bell and can be easily programmed to automatically answer your phone or even to transmit short messages to other machines.

One of the best features provided by D.C. Hayes Associates is the well documented 85 page manual, complete with example programs. However, despite the quality of this manual, there is a glaring omission. I originally purchased the Micromodem II with the notion of easily transfering machine language and BASIC programs to other Apple owners. Although the manual details a procedure for adapting Apple Computer's Datamover program to the Micromodem firmware, easier more direct methods of sending BASIC programs to another computer were not described. This article describes an immediate mode procedure for transfering BASIC programs and also provides an APPLESOFT routine for sending machine language programs or binary data to another Apple II.

Sending a BASIC program in immediate mode is a simple matter using the Micromodem II. Once the phone connection has been established, the receiving computer must be placed in remote mode by sending a CTRL R followed by PR #S where S = modem slot #. When the basic prompt appears, remote control of the Apple at the other end has been achieved. The receiving computer is now waiting input. It will accept commands and input from its own keyboard, your keyboard or those issued automatically by your computer during program execution. In other words, the receiving computer will accept a LISTing of a program sent from another computer and interpret each line as a command. Before LISTing the program, however, a few additonal steps must be taken in order to set up both computers for the transfer.

Once remote control of the receiving machine has been established, the appropriate BASIC must be initialized by typing either the INT or FP DOS command. At this point output from the remote computer should be directed to the video port by executing a PR #0. This is a precautionary step to prevent the accidental transmission of messages generated by the receiving machine's command interpreter. These messages could be received by the sending computer and interfer with the program transfer. The operator of the sending computer will not see the basic prompt return after this command. In order to LIST the program on your computer, terminal mode must be exited by typing CTRL-A/CTRL-X. The receiving

Although this procedure seems complicated, after using it a few times it is easy to remember. For those of you who like to sit back and watch your machine do the work, the following program will create an EXEC file for this purpose.

```
10  REM  BASIC TRANSFER/MICROMODEM II
20  REM  FIRST RUN THIS PROGRAM AND THEN
30  REM  ESTABLISH REMOTE CONTROL OF RECEIVING MACHINE
40  REM  LEAVE TERMINAL MODE BY TYPING CTRL-A/CTRL-X
50  REM  THEN TYPE <EXEC BASIC PROGRAM TRANSFER>
60 D$ =  CHR$ (4)
70  PRINT D$"OPEN BASIC PROGRAM TRANSFER
80  PRINT D$"WRITE BASIC PROGRAM TRANSFER"
90  PRINT "POKE 1530,80:REM FOR LONG FLOATING POINT PRGMS A
      GREATER DELAY MAY BE REQUIRED.
100  PRINT "POKE 1914,18"
110  PRINT "POKE 33,30"
120  PRINT "IN #0"
130  PRINT "PR #2"
140  PRINT "LIST"
150  PRINT "PR #0"
160  PRINT "IN #2"
170  PRINT "TEXT"
180  PRINT "POKE 1530,3"
190  PRINT "POKE 1914,138"
200  PRINT D$"CLOSE"
210  END
```

From now on the commands typed at the local keyboard will not be sent to the remote machine. First, the firmware carriage-return-delay for out-going data must be set by typing POKE 1912 + S,18 followed by POKE 1528 + ,80. The pause after each carriage return allows sufficient time for the receiveing machine to interpret and execute each line before another is sent. Register 1528 + *S* normally contains decimal 3 in terminal mode, which corresponds to a delay of 30 msec. Second, the program to be sent is loaded and the LIST formatting routine disabled by typing POKE 33,30. Finally, a PR #2 is issued and after the cursor returns (0.8 sec), the LIST command given.

Apple is left in remote mode waiting for input, while the sending computer is set up to LIST the program.

Run this program to create the EXEC file, and then LOAD the program you want to send. Finally, EXEC BASIC PROGRA, TRAMSFER. This EXEC file will work with either BASIC. The user's machine will be placed in terminal mode when the transfer is finished. PR #2 must then be issued to the remote computer in order to receive its output.

Binary data or machine language programs can be transmitted in a similar fashion by employing a modified version of the monitor hexadecimal dump routine. Ordinarily upon hitting RETURN this routine displays a hexadecimal address followed by a hyphen following the address. The substitution is necessary because the monitor interpreter requires a colon to immediately follow the address when binary data portion of the F8 ROM chip($FD92-$FDC5) to RAM memory at $1000-$1033. Address $100D was altered from $A0 ("-") to $BA (":"). In addition, the address for the JSR instruction at $1021-1023 was changed from $FD92 to $1000. This HEX dump routine has been incorporated into an APPLESOFT BASIC program which takes care of the housekeeping chores described above for transfering BASIC programs plus a few more. Both the APPLESOFT program and the relocated binary routine at$1000 are listed below.

Although these methods require little software and are easy to implement, they do have a disadvantage. The time required to send BASIC and machine language programs using these techniques is greater (approx. 20 % and 130 %, respectively) than would be expected from the time calculated based upon program length. This is because both INTEGER BASIC and APPLESOFT programs are stored in memory with reserved words tokenized. Tokenized words such as PRINT, POKE, or NEXT require only one byte of memory. Sending a byte at 300 baud takes about 1/30 second, however, with the LISTing procedure described here, transmitting a reserved word such as PRINT requires approximately 5/3o's of a second. Similiarly, with machine language programs, for every 8 bytes of data transfered, a 4 digit hexadecimal address, colon, and 8 pais of hexadecimal data must be sent. A total of 21 characters are sent for every 8 bytes of memory.

In spite of this disadvantage, these techniques are handy for sending medium sized programs over short distances where time is not a costly factor. µ

```
10  REM  BINARY TRANSFER/MICROMODEM II
20 D$ =  CHR$ (4)
30  PRINT D$"NOMON C,I,O"
40  GOSUB 420
50  INPUT "IS RECEIVING COMPUTER IN REMOTE MODE WITH EITHER
     BASIC INITIALIZED ? ";ANS$
60  PRINT
70  IF  LEFT$ (ANS$,1) <  > "Y" THEN  PRINT "TRANSFER
     ABANDONED": END
80  POKE 1530,60: POKE 1914,18: REM  600 MSEC WAIT AFTER
     CARRIAGE RETURN. AUTO LINE FEED IS ACTIVATED AND THE
     WAIT FUNCTION + LOCAL DISPLAY ENABLED.
90  PRINT "STARTING ADDRESS -": INPUT "(MUST END WITH 0 OR
     8) ";ST$
100  REM  LINES 110/170 - HEXADECIMAL TO DECIMAL CONVERSION.
110 Z$ = "0123456789ABCDEF"
120  FOR I =  LEN (ST$) TO 1 STEP  - 1
130  FOR J = 1 TO  LEN (Z$)
140  IF  MID$ (Z$,J,1) <  >  MID$ (ST$,I,1) THEN  NEXT J
150 DEC = DEC + (J - 1) * (16 ^ X)
160 X = X + 1: NEXT I
170 HB =  INT (DEC / 256):LB = DEC - (HB * 256)
180  REM  LINE 190 PLACES THE DECIMAL EQUIVALENTS OF THE
     HIGH & LOW BYTE ADDRESS INTO THE PAGE 0 LOCATIONS USED
     BY THE MEMORY DUMP ROUTINE.
190  POKE 61,HB: POKE 60,LB
200  INPUT "NUMBER OR BYTES (DECIMAL) ";NB
210  PRINT : INVERSE : HTAB 6: PRINT "HITTING ANY KEY
     ABORTS TRANSFER": NORMAL
220  PRINT D$"IN #0"
230  PRINT D$"PR #2"
240  PRINT "CALL -151"
250  PRINT : REM  SENDS CARRIAGE RETURN.
260  FOR I = 1 TO  INT (NB / 8) + 1
270  IF  PEEK ( - 16384) > 127 THEN  POKE  - 16368,0: GOTO
     300
280  CALL 4113: REM  CALLS MACHINE LANGUAGE ROUTINE BELOW.
290  NEXT I
300  PRINT
310  PRINT "3D0G"
320  PRINT D$"PR #0"
330  PRINT
340  POKE 1530,3: REM  NORMAL 30 MSEC WAIT
350  PRINT "              *** ALL DONE ***"
360  PRINT : PRINT "THE SENDING COMPUTER IS NOW IN TERMINAL
     MODE & THE RECEIVING COMPUTER HAS BEEN RETURNED WITH
     BASIC UP IN REMOTE MODE."
370  PRINT : INVERSE : HTAB 15: PRINT "HIT RETURN": NORMAL
380  PRINT D$"IN #2"
390  POKE 1914,138: REM  INITIATE TERMINAL MODE/FULL-DUPLEX
     (USE 10 FOR HALF-DUPLEX).
400  END
410  REM   LINES 420/450 LOAD RELOCATED MEMORY DUMP ROUTINE
     AT $1CCO.
420  FOR M = 4096 TO 4147: READ D: POKE M,D: NEXT M
430  RETURN
440  DATA  164,61,165,60,32,142,253,32,64,249,160,0,169,186,
     76,237,253,165,60,9,7,133,62,165,61,133,63,165,60,41,7,
     208,3,32,0,16
450  DATA  169,160,32,237,253,177,60,32,218,253,32,186,252,
     144,232,96
460  REM   THE BASIC PRGM + DUMP ROUTINE OCCUPY $800-$1040.
     IF THE BINARY DATA TO BE SENT RESIDES IN THIS RANGE,
     IT MUST FIRST BE RELOCATED WITH THE MONITOR MOVE
     COMMAND.
```

NOTE: These Programs were designed for micromodem to reside in slot 2. If another slot is chosen, registers 1530 + 1914 in the page listings must be changed to 1528 + S, respectively where S = Modem slot #

Relocated Monitor Hex Dump

```
1000 A4  3D     LDY $3D
1002 A6  3C  FD LDX $3C
1004 20  8E  Fo JSR $FD8E
1007 20  40     JSR $F940
100A A0  00     LDY #$00
100C A9  BA  FD LDA #$BA
100E 4C  ED     JMP $FDED
1011 A5  3C     LDA $3C
1013 09  07     ORA #$07
1015 85  3E     STA $3E
1017 A5  3D     LDA $3D
1019 85  3F     STA $3F
101B A5  3C     LDA $3C
101D 29  07     AND #$07
101F D0  03  10 BNE $1024
1021 20  00     JSR $1000
1024 A9  A0  FD LDA #$A0
1026 20  ED     JSR $FDED
1029 B1  3C  FD LDA ($3C),Y
102B 20  DA  FC JSR $FDDA
102E 20  BA     JSR $FCBA
1031 90  E8     BCC $101B
1033 60         RTS
```

PET—16

For those PET owners who have envied the Sweet-16 software of the Apple, here is PET-16.

Rev. James Strasma

Fellow PET users, have you envied APPLE people their 16-bit "dream machine"? Now you can have your own PET-16. Converting Apple's Sweet 16 to run on a PET is a fairly simple matter. Three changes are required.

First the program is relocated into user memory.

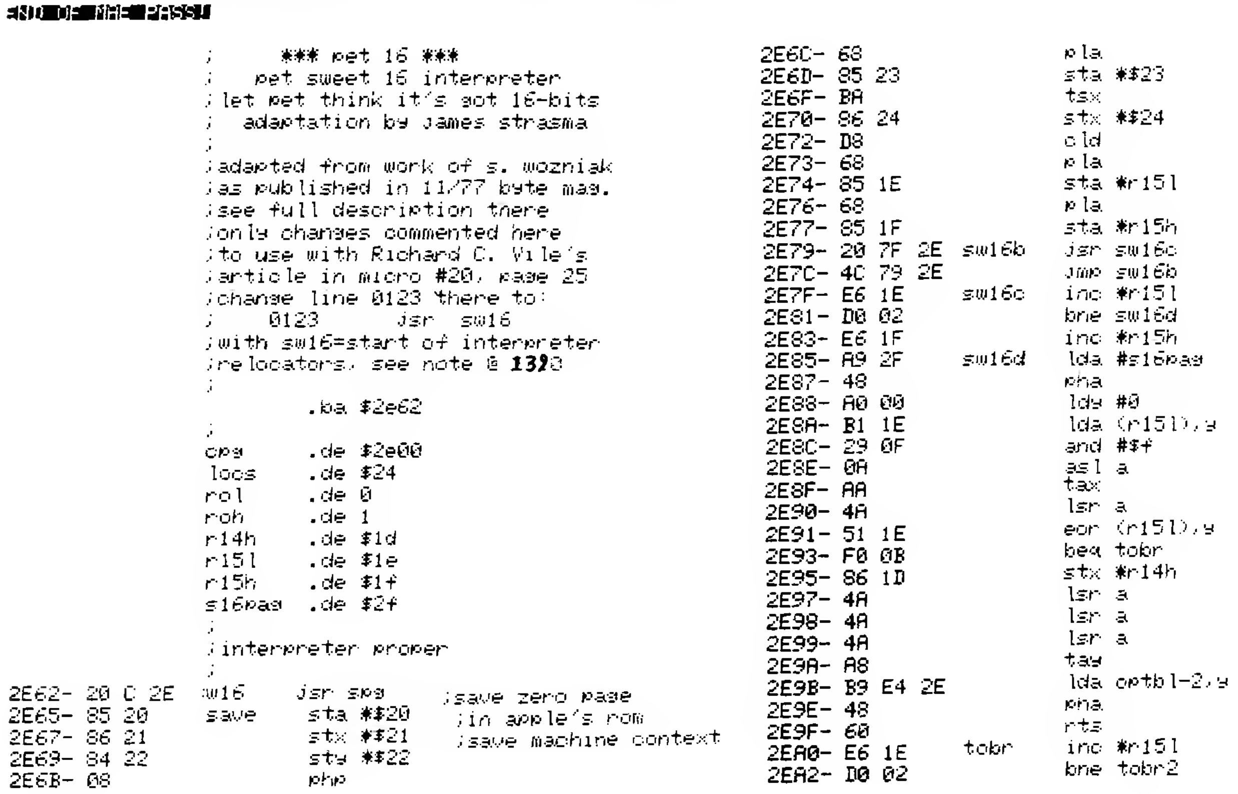
Second, it is altered to exchange a portion of zero page with an unused part of memory. This is necessary because PET Basic and Sweet-16 both use the first 32 memory locations in zero page extensively.

Third, Sweet-16 calls two Apple ROM routines that PET doesn't have. They are 'Save' & 'Restore'. They remember the contents of all the 6502 registers during a program and restore to the original conditions at the end of the program. Thus, this ability must be added to the program. With These few changes, Sweet-16 is quite usable with a PET.

If you have a macro assembler, one key use of PET-16 is in macros. This makes it easier to include simple double-byte routines in programs. Richard C. Vile's article in Micro #20 provides many desirable macros for programming in Sweet-16. Only one line needs to be changed...the actual jsr to Sweet 16 in line 123 of Vile's program. Simply change the destination to the start ot the PET version. This is at $2e62 in the enclosed listing. With that change, the macros are quite usable with PET.

Remember that PET-16 will need to be in memory at the same time as programs that call it. It is just as suitable for placing in a ROM on a PET as on an Apple. That would be the ideal way to keep it handy. Now, who will develop some good uses for PET-16

μ



```
END OF THE PASS!

                  ;       *** pet 16 ***
                  ;     pet sweet 16 interpreter
                  ; let pet think it's got 16-bits
                  ;   adaptation by james strasma
                  ;
                  ;adapted from work of s. wozniak
                  ;as published in 11/77 byte mag.
                  ;see full description there
                  ;only changes commented here
                  ;to use with Richard C. Vile's
                  ;article in micro #20, page 25
                  ;change line 0123 there to:
                  ;     0123      jsr  sw16
                  ;with sw16=start of interpreter
                  ;relocators, see note @ 1320
                  ;
                          .ba $2e62
                  ;
                  cpg     .de $2e00
                   locs   .de $24
                  rol     .de 0
                  roh     .de 1
                  r14h    .de $1d
                  r15l    .de $1e
                  r15h    .de $1f
                  s16pag  .de $2f
                  ;
                  ;interpreter proper
                  ;
2E62- 20 C 2E     sw16    jsr spg      ;save zero page
2E65- 85 20       save     sta *$20     ;in apple's rom
2E67- 86 21                stx *$21     ;save machine context
2E69- 84 22                sty *$22
2E6B- 08                   php
```

```
2E6C- 68                   pla
2E6D- 85 23                sta *$23
2E6F- BA                   tsx
2E70- 86 24                stx *$24
2E72- D8                   cld
2E73- 68                   pla
2E74- 85 1E                sta *r15l
2E76- 68                   pla
2E77- 85 1F                sta *r15h
2E79- 20 7F 2E  sw16b      jsr sw16c
2E7C- 4C 79 2E             jmp sw16b
2E7F- E6 1E     sw16c      inc *r15l
2E81- D0 02                bne sw16d
2E83- E6 1F                inc *r15h
2E85- A9 2F     sw16d       lda #s16pag
2E87- 48                   pha
2E88- A0 00                 ldy #0
2E8A- B1 1E                 lda (r15l),y
2E8C- 29 0F                and #$f
2E8E- 0A                   asl a
2E8F- AA                   tax
2E90- 4A                    lsr a
2E91- 51 1E                eor (r15l),y
2E93- F0 0B                beq tobr
2E95- 86 1D                stx *r14h
2E97- 4A                    lsr a
2E98- 4A                    lsr a
2E99- 4A                    lsr a
2E9A- A8                   tay
2E9B- B9 E4 2E              lda optbl-2,y
2E9E- 48                   pha
2E9F- 60                   rts
2EA0- E6 1E     tobr        inc *r15l
2EA2- D0 02                bne tobr2
```

```
2EA4- E6 1F              inc *r15h
2EA6- BD E7 2E  tobr2    lda brtbl,x
2EA9- 48                 pha
2EAA- A5 1D              lda *r14h
2EAC- 4A                 lsr a
2EAD- 60                 rts
2EAE- 68        rtnz     pla
2EAF- 68                 pla
2EB0- A5 23     restore  lda *$23   ;in apple's rom
2EB2- 48                 pha        ;restore registers
2EB3- A5 20              lda *$20
2EB5- A6 21              ldx *$21
2EB7- A4 22              ldy *$22
2EB9- 28                 plp
2EBA- 20 C0 2E           jsr spg    ;restore zero page
2EBD- 6C 1E 00          jmp (r15l)  ;exchange part of z. p.
2EC0- A2 24     spg      ldx #locs  ;with given area of mem.
2EC2- BD 00 2E  slop     lda cpg,x
2EC5- 48                 pha
2EC6- B5 00              lda *0,x
2EC8- 9D 00 2E           sta cpg,x
2ECB- 68                 pla
2ECC- 95 00              sta *0,x
2ECE- CA                 dex
2ECF- 10 F1              bpl slop   ;max.=$7f locations
2ED1- 60                 rts
2ED2- B1 1E     setz     lda (r15l),y
2ED4- 95 01              sta *roh,x
2ED6- 88                 dey
2ED7- B1 1E              lda (r15l),y
2ED9- 95 00              sa *rol,x
2EDB- 98                 tya
2EDC- 38                 sec
2EDD- 65 1E              adc *r15l
2EDF- 85 1E              sta *r15l
2EE1- 90 02              bcc set2
2EE3- E6 1F              inc *r15h
2EE5- 60        set2     rts
2EE6- 05        optbl    .by set-1
2EE7- FC        brtbl    .by rtn-1
2EE8- 07                 .by ld-1
2EE9- A0                 .by br-1
2EEA- 10                 .by st-1
2EEB- A1                 .by bnc-1
2EEC- 28                 .by ldat-1
2EED- B2                 .by bc-1
2EEE- 19                 .by stat-1
2EEF- B5                 .by bp-1
2EF0- 4A                 .by lddat-1
2EF1- BC                 .by bm-1
2EF2- 54                 .by stdat-1
2EF3- C3                 .by bz-1
2EF4- 32                 .by pop-1
2EF5- CC                 .by bnz-1
2EF6- 5E                 .by stpat-1
2EF7- D5                 .by bm1-1
2EF8- 88                 .by add-1
2EF9- E0                 .by bnm1-1
2EFA- 71                 .by sub-1
2EFB- 08                 .by bk-1
2EFC- 36                 .by popd-1
2EFD- EB                 .by rs-1
2EFE- 73                 .by cpr-1
2EFF- 96                 .by bs-1
2F00- 21                 .by inr-1
2F01- EA                 .by nul-1
2F02- 68                 .by dcr-1
2F03- EA                 .by nul-1
2F04- EA                 .by nul-1
2F05- EA                 .by nul-1
2F06- 10 CA     set      bpl setz   ;from here on must
2F08- B5 00     ld       lda *rol,x ;be in same mem. pg.
                bk       .de =-1
2F0A- 85 00              sta *rol
2F0C- B5 01              lda *roh,x
2F0E- 85 01              sta *roh
2F10- 60                 rts
2F11- A5 00     st       lda *rol
2F13- 95 00              sta *rol,x
2F15- A5 01              lda *roh
```

```
2F17- 95 01              sta *roh,x
2F19- 60                 rts
2F1A- A5 00     stat     lda *rol
2F1C- 81 00     stat2    sta (rol,x)
2F1E- A0 00              ldy #0
2F20- 84 1D     stat3    sty *r14h
2F22- F6 00     inr      inc *rol,x
2F24- D0 02              bne inr2
2F26- F6 01              inc *roh,x
2F28- 60        inr2     rts
2F29- A1 00     ldat     lda (rol,x)
2F2B- 85 00              sta *rol
2F2D- A0 00              ldy #0
2F2F- 84 01              sty *roh
2F31- F0 ED              beq stat3
2F33- A0 00     pop      ldy #0
2F35- F0 06              beq pop2
2F37- 20 69 2F  popd     jsr dcr
2F3A- A1 00              lda (rol,x)
2F3C- A8                 tay
2F3D- 20 69 2F  pop2     jsr dcr
2F40- A1 00              lda (rol,x)
2F42- 85 00              sta *rol
2F44- 84 01              sty *roh
2F46- A0 00     pop3     ldy #0
2F48- 84 1D              sty *r14h
2F4A- 60                 rts
2F4B- 20 29 2F  lddat    jsr ldat
2F4E- A1 00              lda (rol,x)
2F50- 85 01              sta *roh
2F52- 4C 22 2F           jmp inr
2F55- 20 1A 2F  stdat    jsr stat
2F58- A5 01              lda *roh
2F5A- 81 00              sta (rol,x)
2F5C- 4C 22 2F           jmp inr
2F5F- 20 69 2F  stpat    jsr dcr
2F62- A5 00              lda *rol
2F64- 81 00              sta (rol,x)
2F66- 4C 46 2F           jmp pop3
2F69- B5 00     dcr      lda *rol,x
2F6B- D0 02              bne dcr2
2F6D- D6 01              dec *roh,x
2F6F- D6 00     dcr2     dec *rol,x
2F71- 60                 rts
2F72- A0 00     sub      ldy #0
2F74- 38        cpr      sec
2F75- A5 00              lda *rol
2F77- F5 00              sbc *rol,x
2F79- 99 00 00           sta rol,y
2F7C- A5 01              lda *roh
2F7E- F5 01              sbc *roh,x
2F80- 99 01 00  sub2     sta roh,y
2F83- 98                 tya
2F84- 69 00              adc #0
2F86- 85 1D              sta *r14h
2F88- 60                 rts
2F89- A5 00     add      lda *rol
2F8B- 75 00              adc *rol,x
2F8D- 85 00             sta *rol
2F8F- A5 01                lda *roh
2F91- 75 01                adc *roh,x
2F93- A0 00                ldy #0
2F95- F0 E9                beq sub2
2F97- A5 1E       bs       lda *r15l
2F99- 20 1C 2F             jsr stat2
2F9C- A5 1F                lda *r15h
2F9E- 20 1C 2F             jsr stat2
2FA1- 18          br       clc
2FA2- B0 0E       bnc      bcs bnc2
2FA4- B1 1E       br1      lda (r15l),y
2FA6- 10 01                bpl br2
2FA8- 88                   dey
2FA9- 65 1E       br2      adc *r15l
2FAB- 85 1E                sta *r15l
2FAD- 98                   tya
2FAE- 65 1F                adc *r15h
2FB0- 85 1F                sta *r15h
2FB2- 60          bnc2     rts
2FB3- B0 EC       bc       bcs br
2FB5- 60                   rts
```

```
2FB6- 0A             bp       asl a
2FB7- AA                      tax
2FB8- B5 01                   lda *roh,x
2FBA- 10 E8                   bpl br1
2FBC- 60                      rts
2FBD- 0A             bm       asl a
2FBE- AA                      tax
2FBF- B5 01                   lda *roh,x
2FC1- 30 E1                   bmi br1
2FC3- 60                      rts
2FC4- 0A             bz       asl a
2FC5- AA                      tax
2FC6- B5 00                   lda *rol,x
2FC8- 15 01                   ora *roh,x
2FCA- F0 D8                   beq br1
2FCC- 60                      rts
2FCD- 0A             bnz      asl a
2FCE- AA                      tax
2FCF- B5 00                   lda *rol,x
2FD1- 15 01                   ora *roh,x
2FD3- D0 CF                   bne br1
2FD5- 60                      rts
2FD6- 0A             bm1      asl a
2FD7- AA                      tax
2FD8- B5 00                   lda *rol,x
2FDA- 35 01                   and *roh,x
2FDC- 49 FF                   eor #$ff
2FDE- F0 C4                   beq br1
2FE0- 60                      rts
2FE1- 0A             bnm1     asl a
2FE2- AA                      tax
2FE3- B5 00                   lda *rol,x
2FE5- 35 01                   and *roh,x
2FE7- 49 FF                   eor #$ff
2FE9- D0 B9                   bne br1
2FEB- 60             nul      rts
2FEC- A2 18          rs       ldx #$18
2FEE- 20 69 2F                jsr dcr
2FF1- A1 00                   lda (rol,x)
2FF3- 85 1F                   sta *r15h
2FF5- 20 69 2F                jsr dcr
2FF8- A1 00                   lda (rol,x)
2FFA- 85 1E                   sta *r15l
2FFC- 60                      rts
2FFD- 4C AE 2E       rtn      jmp rtnz
                              .en

END OF ONE PASS!!

 ---  LABEL FILE:  ---

add =2F89              bc =2FB3               bk =2F09
bm =2FBD               bm1 =2FD6              bnc =2FA2
bnc2 =2FB2             bnm1 =2FE1             bnz =2FCD
bp =2FB6               br =2FA1               br1 =2FA4
br2 =2FA9              brtbl =2EE7            bs =2F97
bz =2FC4               cps =2E00              cpr =2F74
dcr =2F69              dcr2 =2F6F             inr =2F22
inr2 =2F28             ld =2F08               ldat =2F29
lddat =2F4B            locs =0024             nul =2FEB
optbl =2EE6            pop =2F33              pop2 =2F3D
pop3 =2F46             popd =2F37             r14h =001D
r15h =001F             r15l =001E             restore =2EB0
roh =0001              rol =0000              rs =2FEC
rtn =2FFD              rtnz =2EAE             s16pag =002F
save =2E65             set =2F06              set2 =2EE5
setz =2ED2             slop =2EC2             sps =2EC0
st =2F11               stat =2F1A             stat2 =2F1C
stat3 =2F20            stdat =2F55            stpat =2F5F
sub =2F72              sub2 =2F80             sw16 =2E62
sw16b =2E79            sw16c =2E7F            sw16d =2E85
tobr =2EA0             tobr2 =2EA6
//0000,3000,3000
]
```

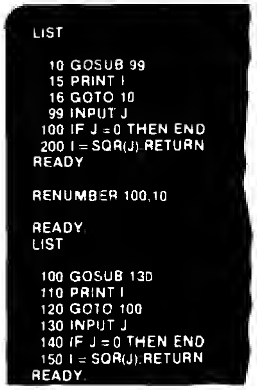
Skyles Electric Works

The BASIC Programmer's Toolkit

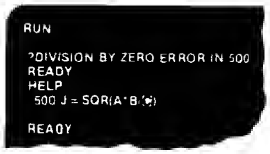
For PET Owners Who Want More Fun And Fewer Errors with Their Programming

Here are Ten Comands you'll need, all on a single chip you can install, in a minute without tools, **on any PET or PET system.** 2 KB of ROM firmware on a single chip with a collection of machine language programs available to you from the time you turn on your PET to the time you shut it off. No tape to load or to interfere with any running programs.

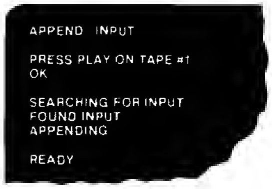
AUTO DELETE RENUMBER HELP TRACE
STEP OFF APPEND DUMP FIND



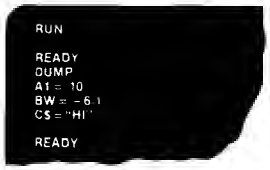
```
LIST

  10 GOSUB 99
  15 PRINT I
  16 GOTO 10
  99 INPUT J
 100 IF J =0 THEN END
 200 I = SQR(J) RETURN
READY

RENUMBER 100,10

READY
LIST

 100 GOSUB 13D
 110 PRINT I
 120 GOTO 100
 130 INPUT J
 140 IF J =0 THEN END
 150 I = SQR(J).RETURN
READY
```



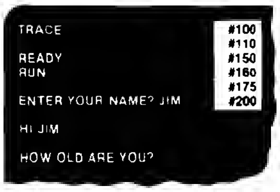
```
RUN

?DIVISION BY ZERO ERROR IN 500
READY
HELP
 500 J = SQR(A*B/(*)

READY
```



```
APPEND  INPUT

PRESS PLAY ON TAPE #1
OK

SEARCHING FOR INPUT
FOUND INPUT
APPENDING

READY
```



```
RUN

READY
DUMP
A1 =  10
BW =  - 6.1
C$ = "HI"

READY
```



```
TRACE                              #100
                                   #110
READY                              #150
RUN                                #160
                                   #175
ENTER YOUR NAME? JIM               #200

HI JIM

HOW OLD ARE YOU?
```

Can be placed in main board socket or with precision-engineered PCB

Now available to interface with your:

PET/CBM models 8N/8B, 16N/16B, 32N/32B (chip only) *TK160N*	$ 50.00*
PET 2001-8 (chip and interface PCB) *TK80P, TK160P*	80.00*
Expandamem *TK80E, TK160E*	80.00*
Skyles Memory Expansion System *TK80S, TK160S*	80.00*
PME 1, RC Factor, Eventide Clockworks *TK80S, TK160S & GTK*	90.00*
Computhink Disk System for PET 1 (original PET) *TK80ED-1, TK160ED-1*	90.00*
Computhink Disk System for PET II (new PET) *TK160ED-2*	100.00*
Commodore Word Pro II (for original PET 2001-8) *TK160 (S or E) & Socket 2 ME*	95.00*
Commodore Word Pro III (for original PET 2001-8) *TK160 (S or E) & Socket 2 ME III*	95.00*
Commodore Word Pro II (for new PETs, CBMs) *TK160N & Socket 2 ME*	72.50*
Commodore Word Pro III (for new PETs, CBMs) *TK160N*	50.00*
Skyles MacroTeA *TK80M, TK160M*	50.00*

**Shipping and handling $2.50 USA, Canada; $10.00 Europe, Asia*

**California residents: please add 6% or 6.5% sales tax as required*

VISA, MASTERCHARGE ORDERS CALL (800) 538-3083 (except California residents)
CALIFORNIA ORDERS PLEASE CALL (408) 257-9140

Skyles Electric Works

231 E South Whisman Road
Mountain View, CA 94041
(415) 965-1735

Slide Show for the SYM

The Apple to SYM Picture Translator permits a SYM with a Visible Memory to use the Apple cassette tapes to put on a 'Slide Show' of its own.

David P.Kemp

Anyone who has visited a computer store recently should be familiar with the program 'Slide Show'. It is a collection of high resolution pictures for the Apple II which occupies two volumes of Apple's user contributed software bank. Photographic images like those of 'Slide Show' are real attention getters and are an excellent demonstration of the capabilities of high resolution graphics. The program described in this article allows the use of Apple Slide Show data to demonstrate the SYM-1/Visible Memory combination.

The visible memory (or K-1008 as it is officially known) is an 8K bit-mapped graphics board for KIM bus computers. It is an almost indespensible addition to the basic SYM-1 as it can be used to perform four distinct functions. As a high resolution graphics device it displays 200 lines by 320 columns for a resolution greater than either the Apple or the PET. As an ordinary 8K block of memory it augments the SYM's limited amount of onboard RAM (4K). As a text display it is faster than a serial terminal, although its maximum density of 25 by 53 characters is inferior to the 25 by 80 format of most terminals. Its most unusual use is that implied by its name. I have found it extremely helpful in debugging a program to be able to see what is going on in memory while the program is executing. This can be done using the visible memory without having to write a single line of special purpose debugging code. By simply placing an I/O buffer in the visible memory address space, for example, it is possibleto see graphically exactly when and how the buffer's contents are changing. The Apple's hires buffer is unsuitable for this purpose because its contents appear on the screen in scrambled order, and the PET and Radio Shack displays cannot be used because they are character oriented. Thus it appears that the visible memory is unique in allowing this very useful mode of operation.

This article is concerned with the visible memory as a pseudo-grey-scale display rather than as a debugging tool. Its manufacturer, Micro Technology Unlimited, sells basic text and graphics subroutines for the board, but there has been little of any published software which makes effective use of it. This program was written to fill a small part of that vacuum. It reads a cassette tape of an Apple hires image and displays it using a portion fo the visible memory.

There are actually two tasks performed by the program: reading data from cassette, and translating the data from Apple high resolution format to a form suitable for use by the visible memory. Because the processor must monitor every bit coming off the tape and because the data translation sometimes requires more than one bit time to complete, the program is a prime candidate for interrupt driven I/O. It could be written without using interrupts by performing the two tasks serially (reading first, then translating) but this approach requires two 8K blocks of memory, which is more than is available on many systems. Interrupts allow both tasks to run simultaneously by making use of processor time that would otherwise be wasted. The cassette reading task is assigned to the interrupt service routine because it must operate in real time if it is to operate at all. The data translation task runs whenever the interrupt routine is not running, ie. it operates in the 'background'.

The most straight forward way of setting up an interrupt structure would be to generate an interrupt request each time the cassette input line changes state. Unfortunately on the SYM this line is connected to a paralled input port where it is not able to trigger an interrupt. The solution to this problem is to use a timed interrupt to enter the service routine and then remain in a wait loop until a tape transition occurs. This technique has the disadvantage of wasting a significant amount of time in the interrupt routine, but it does not matter in this program because the background routine does not require a great deal of time.

The interrupt service routine is responsible for reading bits from cassette, packing them into bytes, and passing the packed data to the background program. It uses two of the SYM's seven programmable timers. One is used to generate interrupts; the other keeps track of the time between tape transitions. If this time is greater than a threshold, the cycle represents a one bit, otherwise it is a zero. The interrupt routine packs the received bits until eight have been accumulated, at which time it stores the packed byte in location BYTE where it can be used by the background program. Because the service routine can be entered at any point in the main program, it must leave all processor registers as it found them. This routine performs bit counting and packing operations directly in memory where they will not interfere with the operation of the main program, thus the accumulator is the only register that must be saved and restored.

Once it has been determined that the cassette input routine will be interrupt driven, the remainder of the program is coded without regard to that fact. In particular, subroutine GETBYT which retrieves data from cassette, contains no explicit timing instructions and just assumes that data will magically appear in memory location BYTE when it becomes available. The background program handles the task of translating image data from Apple format to visible memory format. This includes four distinct operations - reversing bits, packing seven bit bytes into eight bit bytes, unscrambling line numbers, and discarding unused data bytes. The first two operations are handled by subroutine READLN, which reads forty consecutive bytes from tape into thirty five consecutive bytes in the visible memory. The other two operations are handled by subroutine LNADR, which calculates the correct memory address for storing the next line of data or indicates that data is to be ignored. For a complete description of the Apple's convoluted hires memory organization, see MICRO 7:43. The visible memory organization of 320 points by 200 lines is more straightforward. The first forty bytes are displayed on line one, bytes 40-79 on line two, bytes 80-119 on line three, and so on. The bits in each byte appear in order with bit 7 (MSB) to the left and bit 0 (LSB) to the right on the screen. The author's visible memory has been modified to display the maximum 204 lines instead of 200 by altering one trace on the PC board. This modification has no effect on the appearence or organization of the original 200 lines, and since the Apple hires screen contains only 192 lines the modification is not needed to display a full image.

Using the program is a simple matter once some suitable input data is available. This can be obtained using the Slide Show program's write option, or the Apple monitor command 2000. 3FFFW will dump any arbitarary image in the primary hires buffer to tape. (4000.5FFFW will dump the secondary hires buffer.) Play the Apple tape into the SYM and run PICTR at location 200. The image will be loaded into the visible memory in scrambled order as it was placed on the tape, but once the load is complete, the image appears as it did on the Apple screen. The visible memory should be cleared prior to running PICTR because old data will form an undesirable border along the right and bottom margins of the image.

Slide Show images are not the only pictures that can be used with the visible memory of course, but they are the most widely available. Its is possible to digitize almost anything using a graphics tablet, facsimile machine, or TV camera. The author has produced several images for the visible memory in addition to those available for the Apple, but for many users the only source of photographic data is the Apple library. I hope that this program will stimulate the interest of SYM users in both the visible memory and graphic image processing. μ

David Kemp is the owner of an Apple, a SYM, and a Homebrew machine, loosely based on the OSI model 400. His other interests include music, speech, graphics, and interfacing non-standard peripheral hardware. He has developed and is selling an interface to the T.I. 'Speak and Spell', and he is working on other hardware for small systems.

```
          ;***********************************
          ;  APPLE TO SYM PICTURE TRANSLATOR
          ;         D. Kemp   Sept 79
          ;
          ;
          ; This routine reads Apple format cassettes
          ; of high resolution graphics images and
          ; stores the unscrambled data in the K-1008
          ; Visible Memory.
          ;***********************************

00E0      TMP=   $E0                         ;scratchpad
00E1      SCNT=  TMP+1                       ;seven bit counter
00E2      LINE=  SCNT+1                      ;VM line address
00E2      PROD=  LINE                        ;equivalent labels used by MULBYT
00E3      MPCD=  PROD+1
00E4      LINCNT=LINE    +2                  ;line counter
00E5      BYT=   LINCNT+1                    ;interrupt routine bit buffer
00E6      BYTE=  BYT+1                       ;cassette read data
00E7      BCNT=  BYTE+1                      ;interrupt routine bit counter
00E8      LEVEL= BCNT+1                      ;cassette input polarity

8DA9      START= $8DA9                       ;configure for cassette I/O

A000      P1ORB= $A000                       ;VIA #1 base address
A004      P1T1L= P1ORB+4                     ;interrupt timer registers
A005      P1T1H= P1ORB+5
A00B      P1ACR= P1ORB+$B                    ;auxilliary control register
A00D      P1IFR= P1ORB+$D                    ;interrupt flag register
A00E      P1IER= P1ORB+$E                    ;interrupt enable register
A000      TAPIN= P1ORB                       ;cassette input on bit 6

A406      TIMER= $A406                       ;transition timer read register
A415      TIM8=  $A415                       ;transition timer write register
A678      UIRQVC=$A678                       ;user IRQ vector location

0020      VMPAGE=$20                         ;Visible Memory page address

                 *=$200


          ;******************************
```

```
0200  20 C0 02  PICTR  JSR   SYNC        ;set up timer, get cassette header
0203  A2 00            LDX   #0
0205  86 E6            STX   BYTE        ;initialize read data variable
0207  8E 0B A0         STX   P1ACR       ;set interrupt timer one shot mode
020A  8E 05 A0         STX   P1T1H       ;trigger interrupt timer
020D  A9 C0            LDA   #$C0
020F  8D 0E A0         STA   P1IER       ;enable timer interrupts
0212  58               CLI               ;enable IRQs
0213  86 E4     PICT1  STX   LINCNT      ;set line counter
0215  20 2C 02         JSR   LNADR       ;calculate line address
0218  D0 05            BNE   PICT2
021A  20 77 02         JSR   RD8         ;throw away 8 bytes every 3 lines
021D  F0 03            BEQ   PICT3
021F  20 53 02  PICT2  JSR   RDLN        ;read line (40 bytes) into display
0222  A6 E4     PICT3  LDX   LINCNT
0224  E8               INX               ;advance to next line
0225  D0 EC            BNE   PICT1       ;continue if screen not full
0227  8E 0E A0         STX   P1IER       ; else disable interrupts
022A  78               SEI               ;disable IRQs
022B  60               RTS               ; and return to monitor


                ;*****************************
022C  8A        LNADR  TXA               ;scramble line count to produce
022D  0A               ASL   A           ; correct address for loading image
022E  AA               TAX
022F  29 38            AND   #$38        ;performs the following bit mapping
0231  85 E0            STA   TMP         ;  7 -> 2
0233  8A               TXA               ;  6 -> 1
0234  2A               ROL   A           ;  5 -> 0
0235  2A               ROL   A           ;  4 -> 5
0236  2A               ROL   A           ;  3 -> 4
0237  AA               TAX               ;  2 -> 3
0238  29 07            AND   #7          ;  1 -> 7
023A  05 E0            ORA   TMP         ;  0 -> 6
023C  85 E0            STA   TMP
023E  8A               TXA               ; made necessary by the Apple's
023F  0A               ASL   A           ; high resolution hardware
0240  0A               ASL   A
0241  29 C0            AND   #$C0
0243  C9 C0            CMP   #$C0
0245  F0 0B            BEQ   LNUL        ;return z set if line # mod 4 = 0
0247  05 E0            ORA   TMP
0249  A2 28            LDX   #$28        ;multiply scrambled line count by
024B  20 87 02         JSR   MULBYT      ; number of bytes per VM line
024E  09 20            ORA   #VMPAGE     ;***add VM base address***
0250  85 E3            STA   LINE+1
0252  60        LNUL   RTS


                ;*****************************
0253  A0 00     RDLN   LDY   #0
0255  A2 08            LDX   #$8         ;initialize bit count for output byte
0257  A9 07     RDLN1  LDA   #$7
0259  85 E1            STA   SCNT        ;initialize bit count for input byte
025B  20 80 02         JSR   RDBYT       ;get a byte from cassette
025E  4A        RDLN2  LSR   A
025F  26 E0            ROL   TMP         ;reverse bit positions
0261  CA               DEX
0262  D0 0D            BNE   RDLN3       ;check output count
0264  48               PHA               ;done with output word
0265  A5 E0            LDA   TMP
0267  91 E2            STA   (LINE),Y    ;store it in Visible Memory
0269  A2 08            LDX   #$8         ;reinitialize output shift count
026B  68               PLA
026C  C8               INY
026D  C0 23            CPY   #35.
026F  B0 E1            BCS   LNUL        ;exit if line is done
0271  C6 E1     RDLN3  DEC   SCNT
0273  D0 E9            BNE   RDLN2       ;continue if input word not done
0275  F0 E0            BEQ   RDLN1       ;else get another


                ;*****************************
0277  A2 08     RD8    LDX   #$8         ;initialize byte count
0279  20 80 02  RD81   JSR   RDBYT       ;get a byte from cassette
027C  CA               DEX               ; and throw it away
027D  D0 FA            BNE   RD81
027F  60               RTS


                ;*****************************
0280  A5 E6     RDBYT  LDA   BYTE        ;check ready flag
0282  10 FC            BPL   RDBYT       ;wait if no data
0284  46 E6            LSR   BYTE        ;reset flag
0286  60               RTS

```

Classified Ads

```
                        ;*******************************
0287  4A        MULBYT  LSR     A               ;single precision fast multiply
0288  85 E2             STA     PROD            ;enter: A,X = operands
028A  86 E3             STX     MPCD            ;return:
028C  A9 00             LDA     #0              ;       A = unsigned product high
028E  A2 08             LDX     #8              ;    PROD = product low byte  byte
0290  90 03     MULB1   BCC     MULB2
0292  18                CLC
0293  65 E3             ADC     MPCD
0295  6A        MULB2   ROR     A
0296  66 E2             ROR     PROD
0298  CA                DEX
0299  D0 F5             BNE     MULB1
029B  60                RTS


                        ;***** timer interrupt service routine *****
                        ;   assumes no other interrupts are active
029C  48        INT     PHA                     ;save accum
029D  20 E9 02          JSR     GETTR           ;get transition time
02A0  C9 A8             CMP     #$A8
02A2  26 E5             ROL     BYT             ;build byte
02A4  C6 E7             DEC     BCNT            ;check bit count
02A6  D0 10             BNE     INTR            ;not done with byte yet
02A8  A5 E6             LDA     BYTE
02AA  30 13             BMI     ERR             ;overrun if flag not reset
02AC  A5 E5             LDA     BYT
02AE  49 FF             EOR     #$FF            ;correct polarity
02B0  09 80             ORA     #$80            ;set data available flag
02B2  85 E6             STA     BYTE            ;copy data to byte
02B4  A9 08             LDA     #$8
02B6  85 E7             STA     BCNT            ;reinitialize bit count
02B8  A9 00     INTR    LDA     #0
02BA  8D 05 A0          STA     P1T1H           ;retrigger interrupt timer
02BD  68                PLA                     ;restore accum
02BE  40                RTI
02BF  00        ERR     BRK                     ;quit if overrun error


                        ;*******************************
02C0  20 A9 8D  SYNC    JSR     START           ;set up cassette interface
02C3  A2 9C             LDX     #INT
02C5  8E 78 A6          STX     UIRQVC          ;set interrupt vector
02C8  A2 02             LDX     #INT/$100
02CA  8E 79 A6          STX     UIRQVC+1
02CD  A2 08             LDX     #$8
02CF  86 E7             STX     BCNT            ;initialize input bit count
02D1  A2 FA             LDX     #$FA
02D3  8E 04 A0          STX     P1T1L           ;set timer latch for 250 us
02D6  20 E3 02  SYNC1   JSR     GETTRS          ;look for leader
02D9  B0 E5             BCS     SYNC            ;start over if not stable
02DB  CA                DEX
02DC  D0 F8             BNE     SYNC1           ;get enough valid half cycles
02DE  20 E3 02  SYNC2   JSR     GETTRS          ;get sync bit
02E1  90 FB             BCC     SYNC2


                        ;*******************************
02E3  A5 E8     GETTRS  LDA     LEVEL           ;get one half cycle time
02E5  49 40             EOR     #$40
02E7  85 E8             STA     LEVEL

02E9  AD 00 A0  GETTR   LDA     TAPIN           ;get cassette input level
02EC  45 E8             EOR     LEVEL
02EE  29 40             AND     #$40
02F0  F0 F7             BEQ     GETTR           ;wait for end of first half cycle
02F2  AD 00 A0  GETTR1  LDA     TAPIN
02F5  45 E8             EOR     LEVEL
02F7  29 40             AND     #$40
02F9  D0 F7             BNE     GETTR1          ;wait for end of second half cycle
02FB  AD 06 A4          LDA     TIMER           ;get time
02FE  48                PHA
02FF  A9 FF             LDA     #$FF
0301  8D 15 A4          STA     TIM8            ;restart timer
0304  68                PLA
0305  C9 CE             CMP     #$CE            ;get data bit in C
0307  60                RTS

                        .END
```

Hypocycloids on the OSI 540

Here is an update to the Hypocycloids Program which allows it to run on the OSI 540 video board.

E.D. Morris

In the October 1979 issue of MICRO (17:52) I made an offer to supply my Hypocycloids program to owners of OSI 440 video boards. Since then I have received many questions on how to convert this program for use with the newer 540 boards. The following program will draw hypocycloids on a 540 video board; however, the resolution is only 64 x 64. I will supply the program on tape for $3.00 or free if you send me a tape with some of your programs.

Program Notes

Lines 840 to 940 ask for input parameters. For testing, try BIG GEAR = 28 and SMALL GEAR = 7 for a diamond or BIG GEAR = 25 SMALL GEAR = 10 for a five pointed star. Lines 700-760 test if your parameters are reasonable, and, if not, it rejects them. Lines 770-830 calculate points on the hypocycloid curve. The subroutine at line 50 plots a line between OX, OY and NX, NY. This is Bresenham's line drawing algorithm. Note that the subroutine is written in very simple BASIC with no multiplication or division. Thus it can easily (?) be converted to machine code for increased speed. Lines 50-340 determine if the plot should be made using X or Y as an index and changes the sign of variables so that a line can be drawn in any direction. The subroutine from 550 to 630 is specifically for the OSI 540 video board. It turns on a spot at PX, PY. A resolution of 64 in the vertical direction is obtained by turning on half-blocks. The subroutine must decide whether the upper or lower half must be turned on and also not erase a previous half-block. This subroutine can be replaced in other computers by SET PX, PY or PLOT PX, PY if you have either of these commands. μ

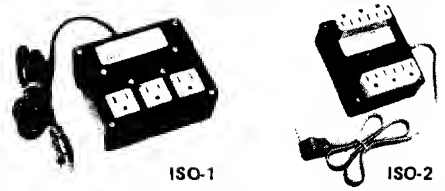
Earl Morris, Jr, a Ph.D. of Physical Chemistry is employed as a research chemist.

He purchased an assembled OSI CPU board about 2 years ago, and built up the remainder of his system from bare boards. He possesses a great interest in hardware and he has made extensive modifications to his own system.

```
10 E=53376
20 S(1,0)=155:S(1,1)=155:S(1,2)=161
30 S(2,0)=154:S(2,1)=161:S(2,2)=154
40 GOTO840
50 SX=NX:SY=NY:FL=0
60 DX=NX-OX
70 IFDX>0THEN90
80 DX=-DX
90 DY=NY-OY
100 IFDY>0THEN120
110 DY=-DY
120 IF DX-DY<0 THEN480
130 REM X AXIS
140 DY=NY-OY
150 IFDY<0THEN250
160 DX=NX-OX
170 IFDX<0 THEN220
180 DD=1
190 X=OX
200 Y=OY
210 GOTO350
220 DX=-DX
230 DD=-1
240 GOTO190
250 DY=-DY
260 DX=NX-OX
270 IF DX<0 THEN320
280 DD=-1
290 X=NX
300 Y=NY
310 GOTO350
320 DX=-DX
330 DD=1
340 GOTO290
350 REM PLOT X AXIS +DY
360 R=DY+DY-DX
370 FORII=0TODX
380 IF FL=1 THENPX=Y:PY=X:GOTO400
390 PX=X:PY=Y
400 GOSUB550
```

```
410 IFR<=0THEN440
420 Y=Y+1
430 R=R-DX-DX
440 X=X+DD
450 R=R+DY+DY
460 NEXTII
470 GOTO530
480 REM Y AXIS
490 T=OX:OX=OY:OY=T
500 T=NX:NX=NY:NY=T
510 FL=1
520 GOTO130
530 OX=SX:OY=SY
540 RETURN
550 M=E+64*INT(PY/2)+PX
560 N=1+(PYAND1)
570 C=PEEK(M)
580 IFC=32 THENL=0:GOTO620
590 IFC=155 THEN L=1:GOTO620
600 IFC=154 THEN L=2:GOTO620
610 RETURN
620 POKEM,S(N,L)
630 RETURN
```

```
640 REM START PROGRAM
700 DT=30:OX=32+P:OY=30:F=6.2832/DT
710 I=1
720 DL=P*I/Q-INT(P*I/Q)
730 IF DL<.0001 THEN760
740 I=I+1:IFI>21 THEN840
750 GOTO720
760 PRINTI:W=P-Q
770 Z=(P-Q)/Q
780 FORJ=0TO I*DT
790 AN=J*F:T=Z*AN
800 NX=32+INT(W*COS(AN)+Q*COS(T))
810 NY=30+INT(W*SIN(AN)-Q*SIN(T))
820 GOSUB50:NEXTJ
830 GOTO830
840 FORJJ=1TO32:PRINT:NEXT
850 PRINT" COMPUTER SPIROGRAPH":PRINT:PRINT:PRINT
860 PRINT:INPUT"SIZE BIG GEAR (20-28)";P
870 IFP>28 THEN860
880 PRINT:INPUT"SIZE SMALL GEAR";Q
890 IFQ>P-4 THEN880
930 FORJJ=1TO32:PRINT:NEXT
940 GOTO700
OK
```



TRACER: A Debugging Tool for the APPLE II

The Apple's Step/Trace routines are handy, but you will find them even more useful when used in conjunction with this Tracer program.

R. Kovacs

Introduction

The APPLE II's monitor in ROM is crammed with many useful routines. These include memory interrogation and modification, keyboard input, Crt display output and cassette I/O. In addition,Apple has thoughtfully provided a number of routines related to assembly language programming. A single-pass assembler and disassembler are invaluable aids in writing and reviewing machine code. A step/trace feature allows the user to control execution of his program during the software development phase.

The step routine executes a single instruction and displays its address, both Hex and disassembled code, the values of the A,X,Y,P registers and the stack pointer. The user has the opportunity to modify any register and continue execution of either the next instruction or any arbitrary one.

Unfortunately, all this information uses up the display rather quickly such that at best only the 11 most recent steps are shown. It seemed to me that it would be useful to display more PC history at the expense of other information.

The Program

The Tracer program was designed to operate in conjunction with APPLE'sstep/trace routines to enhance their usefulness. It is basically a formatter which controls the information output to the screen. This routine will display up to 160 of the most recent instructions executed. This is in addition to the usual

```
                  1000 * TRACER       R. KOVACS   280EC79
                  1010 *
                  1020 *
                  1030 * ENTER VIA CONTROL-Y FOLLOWED BY XXXXT
                  1040 * WHERE XXXX IS THE ADDRESS TO BEGIN TRACING
                  1050 *
                  1060 *******************************
                  1070 *
                  1080 WNDBTM .EQ $23       BOTTOM OF SCROLLING WINDOW
                  1090 PCL    .EQ $3A       PGM COUNTER
                  1100 *
                  1110 WINDOW .EQ $FB3C     SET NORMAL SCROLL WINDOW
                  1120 BELL   .EQ $FBDD     TOGGLE SPEAKER
                  1130 CLEAR  .EQ $FC58     CLEAR SCREEN, HOME CURSOR
                  1140 COUT   .EQ $FDF0     OUTPUT CHAR TO SCREEN
                  1150 READ   .EQ $C000     KEYBOARD STROBE
                  1160 RESET  .EQ $C010     RESET KEYBOARD
                  1170 *
                  1180 BUFF   .EQ $0750     LINE#22-COL#0
                  1190 BUFF1  .EQ $07D0          #23    #0
                  1200 *
                  1210 *******************************
                  1220 *
                  1230 * SET UP CONTROL-Y JUMP TO $3F8
                  1240 *
                  1250        .OR $03F8
                  1260 *
03F8- 4C 00 03    1270        JMP TRINIT
                  1280 *
                  1290 *******************************
                  1300 *
                  1310 * TRACER INITIALIZATION
                  1320 *
                  1330        .OR $0300
                  1340        .TA $5300
                  1350 *
                  1360 *
0300- 20 3C FB    1370 TRINIT JSR WINDOW  CLEAR ENTIRE SCREEN
0303- 20 58 FC    1380        JSR CLEAR
0306- A9 15       1390        LDA #$15    SET SCROLL WINDOW
0308- 85 23       1400        STA WNDBTM
030A- A9 1C       1410        LDA #TRACER SET COUT HOOK
030C- 85 36       1420        STA $36       TO TRACER
030E- A9 03       1430        LDA /TRACER
0310- 85 37       1440        STA $37
0312- A9 1F       1450        LDA #$1F    INIT CH FOR EVEN PAGING
0314- 85 24       1460        STA $24
0316- A9 02       1470        LDA #$02    INIT PGCNT FOR
0318- 8D BD 03    1480        STA PGCNT     SINGLE STEP
031B- 60          1490        RTS
                  1500 *
                  1510 *******************************
                  1520 *
                  1530 *
031C- 8D B7 03    1540 TRACER STA SAVEA   SAVE A & Y
031F- 8C B8 03    1550        STY SAVEY     REGISTERS
0322- 2C BA 03    1560        BIT CRFLG   WAS LAST CHAR A CR?
0325- 30 1C       1570        BMI CR      YES
0327- C9 8D       1580        CMP #$8D    IS THIS CHAR A CR?
```

details (I.E. disassembled code and register displays) of the last instruction displayed. Features include single step and trace with paging. The user can either continue execution or temporarily exit to modify registers or memory. Tracer also looks for the break code (00) and waits for user action after announcing the break with a double bell. The last instruction executed before the break was encountered will still be displayed.

Caution: It should be recognized that Tracer's display lags by one instruction. If the monitor is entered via reset, the current register values saved may be different due to the next instruction having executed. Thus the user should check their values using the control-E monitor command.

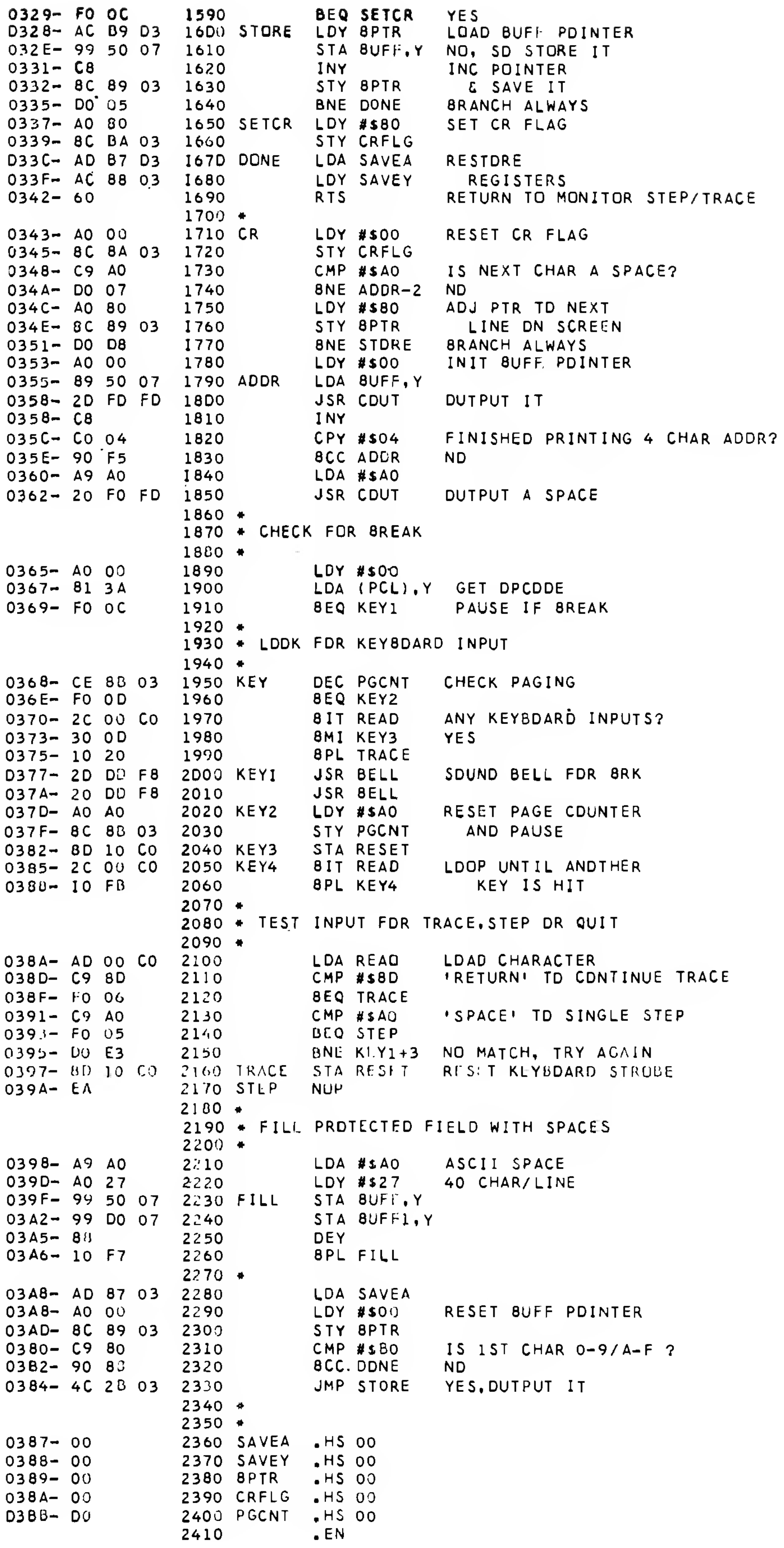
A commented assembly listing is shown in Figure 1. The program is approximately 190 bytes long and is located starting at $300. It uses no additional page zero memory.

How it Works

Tracer controls what information is displayed on the screen by manipulating the characters generated by the step/trace routines. Tracer looks for certain key characters and sequences to determine when one instruction has been completed.

A slight complication arises out of the 2-line display format used by APPLE. The character stream normally output to the screen after completion of a single step begins with a carriage return ($8D). It is then followed by a line of printout whose first 4 characters are the Hex Address of the instruction just executed. This line is terminated with another carriage return and the second line is output.

Tracer looks for the carriage return which marks the beginning of the first line by diverting all characters to Tracer via the COUT hook. Subsequent characters are stored in a buffer. The second line is recoginized by a carriage return followed by a space ($A0). The next carriage return is used tooutput the 4 character Hex address from the buffer (plus a space) to the screen



```
0329- F0 0C             BEQ SETCR   YES
D328- AC B9 D3     STORE  LDY 8PTR    LOAD BUFF PDINTER
032E- 99 50 07          STA 8UFF,Y  NO, SD STORE IT
0331- C8                INY         INC POINTER
0332- 8C 89 03          STY 8PTR      & SAVE IT
0335- D0 05             BNE DONE    8RANCH ALWAYS
0337- A0 80        SETCR  LDY #$80    SET CR FLAG
0339- 8C BA 03          STY CRFLG
D33C- AD B7 D3     DONE   LDA SAVEA   RESTDRE
033F- AC 88 03          LDY SAVEY     REGISTERS
0342- 60                RTS         RETURN TO MONITOR STEP/TRACE
                   *
0343- A0 00        CR     LDY #$00    RESET CR FLAG
0345- 8C 8A 03          STY CRFLG
0348- C9 A0             CMP #$A0    IS NEXT CHAR A SPACE?
034A- D0 07             8NE ADDR-2  ND
034C- A0 80             LDY #$80    ADJ PTR TD NEXT
034E- 8C 89 03          STY 8PTR      LINE DN SCREEN
0351- D0 D8             8NE STDRE   8RANCH ALWAYS
0353- A0 00             LDY #$00    INIT 8UFF PDINTER
0355- 89 50 07     ADDR   LDA 8UFF,Y
0358- 2D FD FD          JSR CDUT    DUTPUT IT
0358- C8                INY
035C- C0 04             CPY #$04    FINISHED PRINTING 4 CHAR ADDR?
035E- 90 F5             8CC ADDR    ND
0360- A9 A0             LDA #$A0
0362- 20 F0 FD          JSR CDUT    DUTPUT A SPACE
                   *
                   * CHECK FOR 8REAK
                   *
0365- A0 00             LDY #$00
0367- 81 3A             LDA (PCL),Y  GET DPCDDE
0369- F0 0C             8EQ KEY1     PAUSE IF 8REAK
                   *
                   * LDDK FDR KEY8DARD INPUT
                   *
0368- CE 8B 03     KEY    DEC PGCNT   CHECK PAGING
036E- F0 0D             8EQ KEY2
0370- 2C 00 C0          8IT READ    ANY KEYBDARD INPUTS?
0373- 30 0D             8MI KEY3    YES
0375- 10 20             8PL TRACE
D377- 2D DD F8     KEYI   JSR BELL    SDUND BELL FDR 8RK
037A- 20 DD F8          JSR 8ELL
037D- A0 A0        KEY2   LDY #$A0    RESET PAGE CDUNTER
037F- 8C 8B 03          STY PGCNT     AND PAUSE
0382- 8D 10 C0     KEY3   STA RESET
0385- 2C 00 C0     KEY4   8IT READ    LDOP UNTIL ANDTHER
0380- 10 FB             8PL KEY4       KEY IS HIT
                   *
                   * TEST INPUT FDR TRACE,STEP DR QUIT
                   *
038A- AD 00 C0          LDA READ    LDAD CHARACTER
038D- C9 8D             CMP #$8D    'RETURN' TD CDNTINUE TRACE
038F- F0 06             8EQ TRACE
0391- C9 A0             CMP #$A0    'SPACE' TD SINGLE STEP
0393- F0 05             BEQ STEP
0395- D0 E3             BNE KEY1+3  NO MATCH, TRY AGAIN
0397- 8D 10 C0     TRACE  STA RESET   RESET KEYBDARD STROBE
039A- EA           STEP   NOP
                   *
                   * FILL PRDTECTED FIELD WITH SPACES
                   *
0398- A9 A0             LDA #$A0    ASCII SPACE
039D- A0 27             LDY #$27    40 CHAR/LINE
039F- 99 50 07     FILL   STA 8UFF,Y
03A2- 99 D0 07          STA 8UFF1,Y
03A5- 88                DEY
03A6- 10 F7             8PL FILL
                   *
03A8- AD 87 03          LDA SAVEA
03A8- A0 00             LDY #$00    RESET 8UFF PDINTER
03AD- 8C 89 03          STY 8PTR
0380- C9 80             CMP #$B0    IS 1ST CHAR 0-9/A-F ?
0382- 90 83             8CC.DDNE    ND
0384- 4C 2B 03          JMP STORE   YES,DUTPUT IT
                   *
                   *
0387- 00           SAVEA  .HS 00
0388- 00           SAVEY  .HS 00
0389- 00           8PTR   .HS 00
038A- 00           CRFLG  .HS 00
D3BB- D0           PGCNT  .HS 00
                          .EN
```

using the monitor COUT routines ($FDF0). These routines take care of wraparound and scrolling to display up to 160 addresses in an 8 by 20 line format.

Since the buffer happens to be part of screen memory, then it too is displayed. The buffer region is protected by moving the bottom of the scrolling window.

The control Y function is used to initialize Tracer via a jump at $3F8. It clears the screen, sets the scrolling window and sets the COUT hook at $36 and $37 to divert all characters normally displayed on the screen to Tracer.

Directions

Tracer is relatively simple to use:

1. Load Tracer starting at $300.
2. Run the program via the monitor by typing: Y(C) XXXX T where Y(C) is a control Y and XXXX is the address where debugging is to begin. The screen will clear, tracer will become hooked via COUT and tracing begins as the specified address.
3. Tracer is initialized to single step and will halt after displaying the familiar step/trace information at the bottom of the screen. Additional steps are executed by depressing the space bar. The addresses of previously executed instructions will begin to accumulate in the upper part of the display.
4. One page (i.e. 160) of instructions can be executed by depressing the return key instead of the space bar. Control can be retained immediately by hitting any key.
5. Of course hitting reset returns the user back to the monitor where registers and memory can be manipulated if needed. Tracer can be reentered by typing: Y(C) T.

µ

Oldest

160 Previously Executed Addresses

Most Recent

Figure 3: **This illustrates Tracer's output format. Example is looping through Apple's WAIT routine at $SCA8. The normal step/trace output for the current instruction is at the bottom of the screen and the previous 160 of program counter are listed above.**

```
WNDBTM 0023    PCL    003A    WINDOW FB3C
BELL   FBDD    CLEAR  FC58    COUT   FDF0
READ   C000    RESET  C010    BUFF   0750
BUFF1  0700    TRINIT 0300    TRACER 031C
STORE  032B    SETCR  0337    DONE   033C
CR     0343    ADDR   0353    KEY    036B
KEY1   0377    KEY2   037D    KEY3   0382
KEY4   0385    TRACE  0397    STEP   039A
FILL   039F    SAVEA  03B7    SAVEY  03B8
BPTR   03B9    CRFLG  03BA    PGCNT  03BB
```

Figure 4: **Symbol Table**

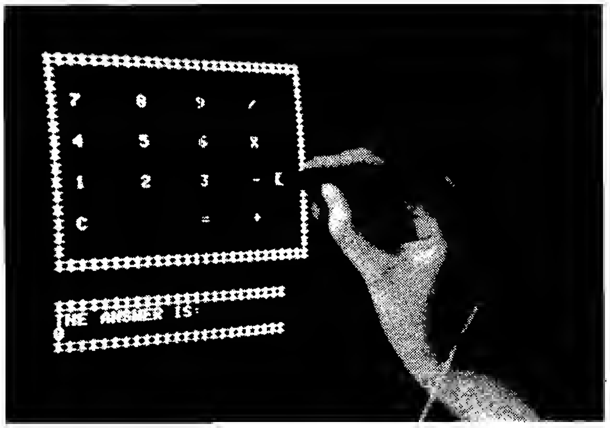
```
FCA9 FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAE
FCAF FCB1 FCA9 FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAE
FCAF FCB1 FCA9 FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAE FCAF FCB1
FCA9 FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAE FCAF FCB1 FCA9 FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAC FCAA FCAC FCAE FCAF FCB1 FCA9 FCAA
FCAC FCAA FCAC FCAA FCAC FCAA FCAC FCAA
FCAA-   E9 01        SBC    #$01
 A=05 X=00 Y=00 P=31 S=99
```

Normal Apple Stop/Trace Display

PROGRESSIVE SOFTWARE

PRESENTS

The LIPSON LIGHT PEN for the APPLE II*



WAS ~~$34.95~~ **$24.95**

Plus $3.00 for postage and handling.
Pa. residents add 6% Sales Tax.

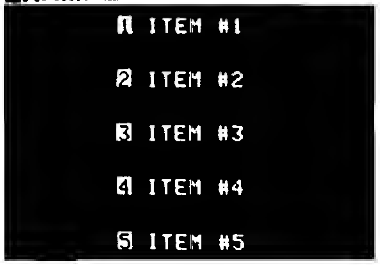
Includes Pen, Disk & Documentation

- Pen plugs directly into Game I/O
- Works with Monitor or TV
- Full 90 Day Guarantee on Pen
- Programs supplied cover all applications needed for the use of the LIPSON LIGHT PEN

The Light Pen includes a total of 15 programs on Disk:

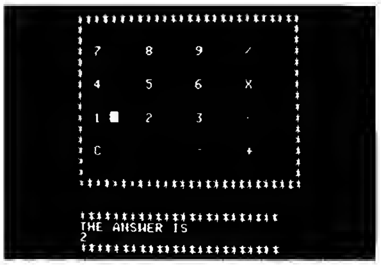
- (2) Menu Selectors - 16K - Integer
- Color Light Pen - 48K - Integer
- Tic - Tac - Toe - 32K - Integer
- Calibrator - 16K - Integer
- Hi - Res Light Graph - 32K - Applesoft
- Pseudo Hi - Res Light Pen - 48K - Applesoft
- Light Pen Calculator - 32K - Integer
- Pen Organ 1. 0 - 16K - Integer
- Music Pen - 16K - Integer
- Hi - Res Light Meter - 32K - Applesoft
- Camera Check - 16K - Integer
- Machine Language Pen Routines - 16K

EXAMPLES:



```
1 ITEM #1

2 ITEM #2

3 ITEM #3

4 ITEM #4

5 ITEM #5
```

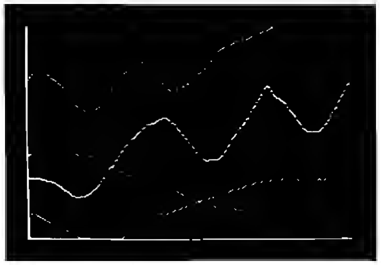
Menu Selector



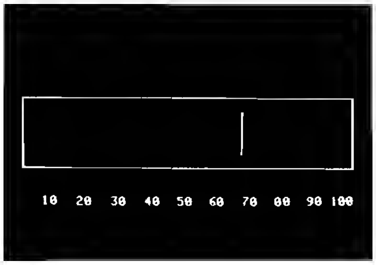
```
7    8    9    /
4    5    6    X
1 ■  2    3    .
C         -    +

THE ANSWER IS
2
```

Light Pen Calculator

Hi - Res Light Graph

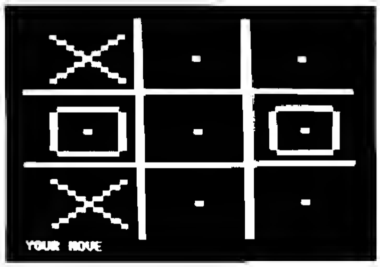


New Hi - Res Color Pen



```
10  20  30  40  50  60  70  80  90  100
```

Hi - Res Light Meter



```
YOUR MOVE
```

Tic - Tac - Toe

To order, send check or money order to:

PROGRESSIVE SOFTWARE

P.O. Box 273
Plymouth Meeting, PA 19462

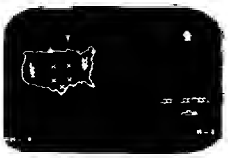
U.S. & Foreign Dealer and Distributor Inquiries Invited.

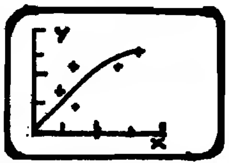
*Apple II is a registered trademark of Apple Computer, Inc.

PROGRESSIVE SOFTWARE

Presents
Software and Hardware for your APPLE

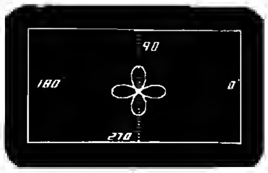
Missile—Anti—Missile (Aplsft) **Curve Fit**





By Dave Garson
Tape-$9.95 Disk-$14.95

Polar Coordinator Plot



by TD Moteles Tape-$9.95 Disk-$14.95

Sales Forecast provides the best forecast using the four most popular forecasting linear regression technques.
Neil D Lipson Tape-$9.95 Disk-$14.95

Single Drive Copy is a utility program, written by Vince Corsetti in Integer BASIC, that will copy a diskette using only one drive. Tape-$19.95 Disk-$24.95

Touch Typing Tutor teaches typing. Indicates speed and errors made. Finger Blds, Gen.Typing, Basic Language and User Supplied. Diskette. Written by Wm. A. Massena. $19.95

Apple Menu Cookbook index-accessed data storage/retrevial program. Recipes stored, unlimited lines per entry. Easy editing. Formulated after N.Y. Times Cookbook. Other useful features included. Written by Wm. Merlino, M.D. $19.95

Mailing List Program maintains complete record of name, address, phone no., mailing lables acommodated parallel card or built-in printer, easy data entry.
Diskette 32K $19.95

Utility Pack combines five versitile programs by Vince Corsetti, for any memory configuration. **Applesoft Update*Integer-to Applesoft conversion * Integer BASIC copy * Binary Copy * Disk Append Tape-$9.95 Disk-$14.95**

Solitare — Old European peg game, played by one (similar to Chinese checkers). Object — to finish with last peg in center. Written by Charles B. Smith Tape-$9.95 Disk-$14.95

Water the Flowers — Math (add., subt., mult., div.,(grades 1-6 (disk). A graphical program that teaches math.
Judy Pegg $19.95

Catch the Pig — Educ. Pkg, 2, An upper grade school game that teaches all four quadrants of the cartesian coordinate system. 4 students play at one time with many levels of play. Aso included is a Linear Version for lower grade school children. Written by Judy Pegg.
Tape-$9.95 Disk-$14.95

Financial Pak — 2. Calculates interest rates on bonds that is based on due date and days between dates. By Neil D. Lipson Disk-$14.95

- **Programs accepted for publication**
- **Highest royalty paid**

U.S. and foreign dealer and distributor inquires invited
All programs require 16K memory unless specified

☆ ☆ ☆ FILES ☆ ☆ ☆

*Builds Serial Files
*Changes Serial Files to random access Files
*Adds to End of Serial Files
*Record insertion and deletion anywhere in Serial File.
*Move individual records or blocks of records within Serial Files

A File manipulator that allows the user to search for a string within a file, sort date by blocks handle many files at a time (without exiting the program and saves executed files. A file can be saved under many names, viewed in several modes, and dumped (totally or partially to a printer).

If you would like to or work with files you must own this program! Comes with 10-page doumentation in a binder. Because of the size and weight of this program postage and handling charge is necessary.
FILES-Disk only.$49.95 plus $4.95 postage and handling.
Written by Mare Gold Farb

☆ ☆ ☆ ROSTER ☆ ☆ ☆

*A complete package for Educators! Roster is a general purpose disk-based record-keeping program for teachers at all levels. It allows instructors to create and change class rosters label, enter and change tast or assignment scores, sort the roster based on student number, student name, or rank in class, assign character or numeric grades based on any of five criteria (raw score, percent, rank percantile rank or Z-score) and Lists scores, totals (or averages), and /or grades according to any of these options.
Roster on Disk (only) $49.95 plus $4.95 Postage and Handling.
Written by Douglas B. Eamon, Ph.D.

Hardware

Light Pen with seven supporting routines. Some of these are light meter, light calculator, Light pen, ;and Light pen TIC TAC TOE. The light pen connects points in high or low resolution graphics. Neil Lipson's program uses artificial intelligence; the pen is not confused by outside light. Requires 48K and Applesoft in ROM. $24.95
Plus $3.00 Postage and handling.

TO ORDER

Send Check or Money Order to:

P.O Box 273
Plymouth Meeting, PA 19462

PA residents add 6% sales tax.

POSTAGE AND HANDLING

Please add $1.50 for the first item and $1.00 for each additional item.

Stop That PET - Update

Program updates to 'Stop That PET' for the new ROMs

The assembler file listing can be used to convert the program **'Stop That PET'** by Gary Bullard in MICRO 22:57 for use with the new ROM PETs. The changed addresses are contained in lines 130 thru 200 or those labels defined as external.

The new PETs will go bye-bye on endless loops. The modifications have been tested on several loops and work as Mr. Bullard's article says they should.

µ

George R. Gaukel
335 ASA Co., Box 63
Ft. Lewis, WA 98433

```
0100;RESET
0110;
0120;
0130VECTOR         .DE $0090
0140KNT            .DE $009E
0150KBUF           .DE $026F
0160BASIC          .DE $C389
0170ASPR           .DE $CA1C
0180DCPR           .DE $DCD9
0190CONINT         .DE $E62E
0200STOP           .DE $F301
0210;
0220               .BA $1F40
0230               .OS
0240               .CE
0250;
0260START          SEI
0270               LDY VECTOR+1
0280               JSR SET
0290               CPY #$E6
0300               BNE NOTSET
0310               JSR RESET
0320NOTSET         CLI
0330               RTS
0340;
0350CKSTOP         JSR STOP
0360               BEQ STOPPD
0370               JMP CONINT
0380STOPPD         LDX #$08
0390SAVEM          PLA
0400               STA STK-1,X
0410               DEX
0420               BNE SAVEM
0430;
0440RESTK          LDA #H,BASIC
0450               PHA
0460               LDA #L,BASIC
0470               PHA
0480               LDA STATUS
0490               PHA
0500               PHA
0510               PHA
0520               PHA
0530               JSR SET
0540;
0550               LDX #$09
0560               STX KNT
0570SETSYS         LDA SYS66-1,X
0580               STA KBUF-1,X
0590               DEX
0600               BNE SETSYS
0610               JMP CONINT
0620;
0630DISPLAY        LDY #H,HEADER
0640               LDA #L,HEADER
0650               JSR ASPR
0660               LDA LOC
0670               LDX LOC+1
0680               JSR DCPR
0690               LDY #0
0700               STY CNTR
0710ALLREG         LDY CNTR
0720               CPY #4
0730               BEQ START
0740               LDX STATUS,Y
0750               LDA #0
0760               JSR DCPR
0770               INC CNTR
0780               JMP ALLREG
0790;
0800               NOP
0810               NOP
0820;
0830SET            LDA #H,CONINT
0840               LDX #L,CONINT
0850SAVIT          STA VECTOR+1
0860               STX VECTOR
0870               RTS
0880RESET          LDA #H,CKSTOP
0890               LDX #L,CKSTOP
0900               JMP SAVIT
0910;
0920;
0930SYS66          .BY $91
0940               .BY $53
0950               .BY $59
0960               .BY $53
0970               .BY $39
0980               .BY $70
0990               .BY $36
1000               .BY $36
1010               .BY $0D
1020               NOP
1030;
1040;
1050HEADER         .BY $91
1060               .BY $20
1070               .BY $41
1080               .BY $44
1090               .BY $44
1100               .BY $52
1110               .BY $20
1120               .BY $53
1130               .BY $54
1140               .BY $20
1150               .BY $41
1160               .BY $43
1170               .BY $20
1180               .BY $58
1190               .BY $52
1200               .BY $20
1210               .BY $59
1220               .BY $52
1230               .BY $0D
1240               .BY $00
1250;
1260               .BA $1FE5
1270;
1280STK            .BY $00
1290               .BY $00
1300LOC            .BY $00
1310               .BY $00
1320STATUS         .BY $00
1330               .BY $00
1340               .BY $00
1350               .BY $00
1360CNTR           .BY $00
1370               .EN
//
```

```
LABEL FILE: [ / = EXTERNAL ]
/VECTOR=0090            /KNT=009E
          /KBUF=026F
/BASIC=C389             /ASPR=CA1C
          /DCPR=DCD9
/CONINT=E62E            /STOP=F301
          START=1F40
NOTSET=1F4E             CKSTOP=1F50
          STOPPD=1F58
SAVEM=1F5A              RESTK=1F61
          SETSYS=1F76
DISPLAY=1F82            ALLREG=1F97
          SET=1FAE
SAVIT=1FB2              RESET=1FB9
          SYS66=1FC0
HEADER=1FCA             STK=1FE5
          LOC=1FE7
STATUS=1FE9             CNTR=1FED

//06FF,0200,0200
```

6502 Resource Update

An list of magazines which contain information about the 6502 on a reasonably regular basis.

Dr. William R. Dial

Did you ever wonder just what magazines were rich sources of information on the 6502 microprocessor, 6502-based microcomputers, accessory hardware and software? For several years this writer has been assembling a bibliography of 6502 references related to hobby computers and small business systems. The accompanying list of magazines has been compiled from this bibliography. An attempt has been made to give up-to-date addresses and subscription rates for the magazines cited. Subscription rates are for the U.S. Rates to other countries normally are higher.

GENERAL 6502

MICRO
$15.00 per year,12 issues
MICRO
P.O. Box 6502
Chelmsford, MA 01824
(Includes OSI Small Systems Journal)

COMPUTE
(PET,Atari, Apple)
$9.00 per year,6 issues
COMPUTE II
(AIM, SYM, KIM, OSI, ...)
$9.00 per year,6 issues
Compute
Small System Services, Inc.
900-902 Spring Garden Street
Greensboro, NC 27403
(Absorbed PET Gazette, 6502 Users Notes, and others)

GENERAL COMPUTER

BYTE
$18.00 per year,12 issues
Byte Publications, Inc.
70 Main Street
Peterborough, NH 03458

COMPUTER CASSETTES REVIEW
$12.00 per year,quarterly
Robert Purser
P.O. Box 466
El Dorado, CA 95623

COMPUTER SHOPPER
$10.00 per year
Glenn Patch, Editor
P.O. Box F
Titusville, FL 32780
(Has absorbed ON_LINE)

CREATIVE COMPUTING
$14.00 per year,12 issues
Creative Computing
P.O. Box 789-M
Morristown, NJ 07960

DIGITAL DESIGN
$20.00 per year
Benwill Publishing Corp.
1050 Commonwealth Avenue
Boston, MA 02215

DR. DOBB'S JOURNAL
$15.00 per year,10 issues
Peoples Computer Co.
Box E
1263 El Camino Real
Menl Park, CA 94025

EDN(Electronic Design News)
$25.00 per year,22 issues
Cahners Publishing Co.
270 St. Paul Street
Denver,CO 80206

INTERFACE AGE
$18.00 per year,12 issues
McPheters, Wolfe & Jones
16704 Marquardt Avenue
Cerritos, CA 90701

KB MICROCOMPUTING
$18.00 per year,12 issues
Microcomputing
Pine Street
Peterborough, NH 03458

ON COMPUTING
$8.50 per year,quarterly
P.O. Box 307
Martinville, NJ 08836

PERSONAL COMPUTING
$14.00 per year,12 issues
Benwill Publishing Corp.
1050 Commonwealth Avenue
Boston, MA 02215

POPULAR COMPUTING
$18.00 per year,12 issues
P.O.Box 272
Calabasas, CA 91302

RECREATIONAL COMPUTING
$10.00 per year,6 issues
Peoples Computer Co.
1263 El Camino Real
Box E
Menlo Park, CA 94025

THE ABACUS II NEWSLETTER
$12.00 per year,12 issues
2850 Jennifer Drive
Castro Valley, CA 94546

APPLE
$2.00 per issue, quarterly
Apple Computer Co.
10260 Bandley Drive
Cupertino, CA 95014

APPLE BARREL
Ed Seeger
4331 Nenana Drive
Houston, TX 75006

APPLE BITS
$14.00 per year,$2.00 Apple fee
NEO Apple Corps
John D. Ross
31900 N. Marginal Road
Apartment 522
Willowick, OH 44121

APP-LE-CATIONS
Conrad P. Pracht
5101-140 Park Road
Charlotte, NC 28209

APPLE-COM-POST
(Apple User Group Europe)
Postfach 4068
D-4320 Hattingen
West Germany
(Printed in German)

APPLE GRAM
$10.00 per year,12 issues
Apple Corps of Birmingham
Gerald C. Jenkins
774 Twin Branch
Birmingham, AL 35226

APPLE GRAM
$12.00 per year,12 issues
Apple Corps of Dallas
Bobbie Ferrell
15255 Midway Road
Dallas, TX 75240

APPLE ORCHARD NEWSLETTER
$12.00 per year,6 issues
131 Highland Avenue
Vacaville, CA 95688

APPLE PEEL
$10.00 per year,12 issues
Apple corps of Birmingham
Gerald C. Jenkins
774 Twin Branch
Birmingham, AL 35226

APPLESAUCE
$10.00 per year,12 issues
12804 Magnolia
Chino, CA 91710

APPLESEED
Bill Hyde
The Computer Shop
6812 San Pedro
San Antonio, TX 78216

THE APPLE SHOPPE
$12.00 per year,8 issues
P.O. Box 701
Placentia, CA 92670

CALL A.P.P.L.E.
$15.00 per year
($25.00 application fee)
517 11th Avenue E.
Seattle, WA 98102

THE CIDER PRESS
$15.00 per year,12 issues
San Francisco Apple Core
P.O. Box 4816
San Francisco, CA 94101

FROM THE CORE
$12.00 per year, 12 issues
Carolina Apple Core
P.O.Box 31424
Raleigh, NC 27612

FWAUG
$15.00 per year, 12 issues
Fort Worth Apple User Group
1401 Hillcrest Drive
Arlington, TX 76010
Lee Meador, Editor

HARVEST
$12.00 per year
No. West Suburban Apple User Group
650 Pompano Lane
Palatinine, IL 60067

MIN 'APP'LES NEWSLETTER
$10.00 per year
Mini'App'Les Apple Computer
User Group
Keith Madonma
23885 Clowel Lane
Excelsior, MN 55331

NEAT NOTES
$6.00 per year, 12 issues
New England Apple Tree
Mitch Kapon
31 Birch Road
Watertown, MA 02172

NEWSLETTER
$10.00 per year
Apple Bytes of Buffalo
Hank Kolk
171 Tree Haven Road
Buffalo, NY 14215

NIBBLE
$15.00 per year,8 issues
S.P.A.R.C.
P.O. Box 325
Lincoln, MA 01773

THE ORCHARD
$1.00 per issue to member clubs
Only 1 issue published to date, noted inside that there may or may not be more issues. Scheduled to contain CONTACT.
International Apple Core
P.O. Box 976
Daly City, CA 94017

POKE APPLE
$10.00 per year, 12 issues
Applesiders
5707 Chesapeake Way
Fairfield, OH 45014

RAINBOW
$15.00 per year, 12 issues
P.O.Box 43
Audubon, PA 19407

RUBBER APPLE NEWSLETTER
$12.00 per year, 10 issues
J. Scotty Musgrave
203 17th Street N.W.
Barberton, OH 44203

THE SEED
$12.00 per year, 12 issues
P.O.Box 17467
Denver, CO 80217

SOFTSIDE
$15.00 per year
P.O.Box 68
Milford, NH 03055

SOUTHEASTERN SOFTWARE NEWSLETTER
$10.00 per year,10 issues
George McClelland
7270 Culpepper Drive
New Orleans, LA 70126

WASHINGTON APPLE PI
$12.00 per year, 12 issues
P.O.Box 34511
Washington, DC 20034

AIM

INTERACTIVE
$5.00 for 6 issues
Newsletter Editor
Rockwell International
P.O. Box 3669, RC55
Anaheim, CA 92803

THE TARGET
$5.00 per year, 6 issues
Donald Clem, Editor
RR#2
Spencerville, OH 45887

OSI

OSI USER'S INDEPENDENT NEWSLETTER
$10.00 per year, 6 issues
Charles Curley
6061 Lime Avenue #2
Long Beach, CA 90805

PEEK (65) OSI USER JOURNAL
$8.00 for 12 issues
62 Southgate Avenue
Annapolis, MD 21401

PET

THE PAPER
$15.00 per year,10 issues
The Paper
P.O. Box 43
Audobon, PA 19407

SYM

SYM-PHYSIS
$9.00 per year, 6 issues
P.O.Box 315
Chico, CA 95927

Non-Computer Magazines

POPULAR ELECTRONICS
$14.00 per year, 12 issues
One Park Avenue
New York, New York 10016

QST
$18.00 per year, 12 issues
American Radio Relay League
225 Main Street

RADIO ELECTRONICS
$13.00 per year, 12 issues)
Gernsback Publications, Inc.
200 Park Avenue, South
New York, New York 10003

73 MAGAZINE
$18.00 per year, 12 issues
73, Inc.
Peterborough, NH 03458

μ

[Editors' Note: A number of the smaller independent magazines have disappeared during the past year. Some have been incorporated in other journals - some have not.]

[Editor's Note: If your publication is not listed, and you think that it should be, then please put Dr. Dial on your subscription list so that you may be covered in the continuing 6502 Bibliography, and start an exchange subscription with us here at MICRO.]

CP/A

OPTIMIZED SYSTEMS SOFTWARE

PRESENTS

CONTROL PROGRAM/APPLE
the DOS you have been waiting for

OSS CP/A is an all new, disk-based operating system which provides commands and utilities similar to CP/M®. CP/A has byte and block I/O, a simple assembly language interface, and direct access via Note and Point. And it's easy to add your own commands or device handlers. CP/A is expandable, flexible, consistent, easy-to-use and available now with compatible program products:

BASIC — Some of the features of OSS BASIC are syntax checking on program entry, true decimal arithmetic (great for money applications), 32K byte string sizes, flexible I/O, long variable names (up to 255 significant characters), and the ability to get and put single bytes.

BUSINESS BASIC WITH PRINT USING — This is virtually the only basic available on the Apple that has PRINT USING. It also has record I/O statements and all the features of our standard BASIC.

EDITOR/ASSEMBLER/DEBUG — OSS EASMD is a total machine language development package. The editor provides functions like FIND, REPLACE, etc. The assembler uses standard 6502 mnemonics, can include multiple files in one assembly, and can place the object code in memory or to a disk file.

Prices of CP/A with:

BASIC	$ 69.95
Business BASIC	84.95
EASMD	69.95
BASIC + EASMD	109.95
Business BASIC + EASMD	124.95

Requires 48K RAM and DISK

Add $3.50 for shipping and handling in continental USA. California residents add 6%. VISA/Master Charge welcome. Personal checks require two weeks to clear.

SEE YOUR DEALER or ORDER TODAY

OPTIMIZED SYSTEMS SOFTWARE
Is a product of
Shepardson Microsystems, Inc.
20395 Pacifica Dr., Suite 108B
Cupertino, CA 95014
(408) 257-9900

THE BASIC PROBLEM

~FILES~

the answer,

Simpl-A-filer®

- * System - Filer, Editor
- * Print-using Subroutine
  (Add to your Basic programs)
- * Interactive File System -
  with print-using formatting
- * Full Printer Capability
- * Text File - Copying, Assembly

Dealer Inquiries Invited

ONLY

$ 50

Penna. resident please add 6% sales tax.

DESIGNED FOR THE

apple® computer

*Apple is a Registered TM of Apple Computers, Inc.

To Order: Send check or money order to

(215) 386-7994

INI Inc.

4029 Spruce Street Box 15
Philadelphia, PA 19104

MICRO Club Circuit

Here is another installment of 6502-related clubs. The response to MICRO's effort to update this section has been terrific! We hope that all 6502 clubs will soon be presented here at some point.

This is a list of some of the newer clubs to have registered with MICRO, or that have sent us an update. If your club has been active, why not let us know what it has been up to?

If you are a newly formed group, have your representative register your group with us. In return for this registration we will send you a free one year subscription to MICRO for your club's library. Include information regarding your club's name, location, algorithm, publications, purpose, officers, membership total, contact person, and/or any other information you would like to share with the world! Your club will then automatically appear in any club update. If you are already registered, don't forget to keep MICRO up-to-date.

Address any correspondence to:
MICRO CLUB CIRCUIT
P.O. Box 6502
Chelmsford, MA 01824

If any information presented here is incorrect or outdated, please send us the correction to be made. We will remedy the situation in the next possible issue. We are striving to make the Club Circuit as helpful as possible.

Tulsa Apple User Group

Meets on second Tuesday of each month (7:30 p.m.) at:

High Technology Store
Tulsa, Oklahoma

John Shanks is the President of this group of 40 users. This group aims to exchange educational information. Contact:

Bill Dufresne
Tulsa Computer Society
P.O.Box 1133
Tulsa, OK 74101

MINI'APP'LES

Meets on the third Wednesday of each month (7:30 p.m.) at:

Federal Savings & Loan
9th Avenue S
Hopkins, Minn.

Dan Buchler is President of this 150+ group and can be contacted at:

13516 Grand Avenue S
Burnsville, MN 55337

"We are now 2 years old and growing strong!"

THE APPLE CART

This is an international membership club which keeps in contact through a bi-monthly newsletter. *"Some objectives: Provide members with timely information about Apple Hardware and Software; provide members with a forum to share their experiences with and uses of their Apples; provide access to quality software by maintaining a software exchange; and to promote the creation of well written and well documented software."* Please address any letters (only) to:

C. Brandon Gresham, Jr.
Bin "R", Project 5810-1
Pasadena, CA 91109

MICROCOMPUTER INVESTORS ASSOCIATION

Meets as called at:

902 Anderson Drive
Fredericksburg, VA 22401

Jack Williams is the Administrator and can be contacted at the above address.

"Using microcomputers to make and manage investments."

Lincoln Computer Club

Meets as needed to schedule computer use and business. John Fultz is the advisor for this group. He may be contacted at:

Lincoln School
750 E. Yosemite Avenue
Manteca, CA 95336

"Educational aims. Group is made up of 7-8 students who use the computers before and after school."

Apple Creek

This is a new name for the Apple User's Group listed last month. They inform us that they have no dues, norany officers but every third Thursday, 40 people show up to hear scheduled speakers. They are still at:

Computer Land of Walnut Creek
1815 Ygnacio Valley Road
Walnut Creek, CA 94598

Apple-Dayton

Here is another update for this club. It now meets on the second Thursday of each month (7:30 p.m.) at:

Bldg. 640
Wright-Patterson AFB

Bob Rennard, President. To contact this club, write to:

Dick Peschke, Sec.
4819 Leafburrow Drive
Dayton, Ohio 45424

"To acquire, distribute and organize programs to benefit members and to promote informed use of the Apple II systems in Dayton are some of thisclub's goals."

Iowa City Apple User's Group

Meets the third Tuesday of every month (7:30 p.m.) at:

Westinghouse Data-Score
Systems I-80 and IA-1

Contact David B. Thomas for more information:

134 Ravencrest Drive
Iowa City, IA 52240

The Boston Computer Society

Meets the fourth Wednesday of each month (except in July). Their mailing address is:

17 Chestnut Street
Boston, MA 02108

Jonathan Rotenberg is this club's President. Membership is expected to exceed 700 by this month.

"A total microcomputer resource center offering seven user groups, four publications, plus a variety of special events including seminars and general meetings with top authorities from around the world. Special groups dedicated to PASCAL and micros in education, also."

The Computer Network of K.C.

Meets on the second Sunday of each month (7:30 p.m.) at:

425 Volker Boulevard
Kansas City, Kansas

George Schiell is the Club President. Membership is currently at 90. For further information, contact:

Harold J. Schwartz
1251 Kansas Avenue
Kansas City, Kansas 66105

"EMS and Hardware and software. Not a TRS 80 Group."

OSIO
National Headquarters

Meets on the first Tuesday of each month (7:30 p.m.) at:

Walter Johnson High School
Rockville, MD.

Wallace Kendall is President of this club which consists of 215 members, and still growing. National Headquarters is located at:

9002 Dunloggin Road
Ellicott City, MD. 21043

You may write them at the above address for further information.

"Encouraging local chapters. Study and promote public understanding of small computers. Our interest is centered on OSI machines. Active exchange program using data disks (no op systems or proprietary software). Starting an on-line hard-disk system which will permit exchange by phone. Several discount arrangements. OSIO is incorporated in the District of Columbia as a non-profit educational organization. We will be sponsoring seminars, tutorials, etc."

Kalamazoo Apple Computer User Society

Meets on the third Thursday of every month (7:30 p.m.) at:

Computer Room
455 W. Michigan Avenue
Kalamazoo, MI 49007

Gary Wilkins is President for this club of twenty-four. For information, contact Alex Ellingsen at the above address, or:

Gary Wilkins
3606 Thornhill Avenue
Kalamazoo, MI 49007

"Aid new and old Apple users with the full benefits and use of the Apple Computer."

Amateur Radio Research and Development Corp.

Meets on the first Monday of each month (7:30 p.m.) at:

Patrick Henry Branch Library
Vienna, VA 22180

Paul L. Rinaldo is President to this group of 260. You may write him for more information, at:

1524 Springvale Avenue
McLean, VA 22101

"Technical interests of ham radio and computing. Project in deaf communications."

Northwest Suburban Apple Users Group (NSAUG)

Meets at the Schaumburg Library in IL. Meetings are held on the first Saturday of each month. Don Fuller is President and can be written to at:

1140 Old Mill Drive
Palatine, Il 60067

"Promote knowledge, information and interest regarding the Apple II Computer. 135 members currently."

M^{3}C^2
Mid Michigan Micro Computer Club

Meets on the second Monday of each month at 7:30 p.m. Boasts of 103 members. For more information regarding this club which is open for all types of Microcomputers, please write to:

Earl Morris
3200 Washington
Midland, MI

"Meetings held at Delta College with the room numbers announced in each month's newsletter. Membership is informal and open to all interested in computers. Ages range from 10 to 65 and from novice to expert. Feature demonstrations of equipment and programs."

New England Apple Tree

This group meets on the third Wednesday of each month at 7:30 p.m. at the Mitre Corp. in Bedford, MA. Richard Sedgewick, President. More information can be requested by writing to:

Ruth Souza, Sec.
P.O. Box 2652
Woburn, MA 01888

"We publish a newsletter to keep our members up-to-date."

Fort Worth Apple Computer User Group

This club meets on the third Sunday of each month at 3:00 at either:

Micro Age Computer
1220 Melbourne
Hurst, TX

or

CompuShop
6353 Camp Bowie
Ft. Worth, TX

Marshall Martin is President to about 40 members. For current information, contact:

Lee Maudor
FWAUG
1401 Hillcrest Drive
Arlington, TX 76010

"We aim to provide a forum for exchange of information between users of Apple computers and to provide a medium for personal growth in programming, writing, design, and similar areas."

Dayton Microcomputer Association

The last Tuesday of each month is when this club gathers, at 7:30 p.m. Club location is at

2629 Ridge Avenue
Dayton, OH 45414

David Taylor is President for about 100 members. He may be written to at:

259 Aberdeen Avenue
Dayton, OH 45419

"Our purpose is to promote communication among computer hobbyists."

KIM Users Club
The Netherlands

Meets on the third Saturday of every odd month, except July. The club's location is at:

Sinjeur Semeynsstr
78I 1061 GM Amsterdam

For further information write to the club secretary, Anton Muller, at the above address.

"Knowledge and experience exchange on any 65XX based systems."

TRACE
Toronto Region Assoc. of ComputerEnthusiasts

Meets at different times during the month at either the Ontario Science Centre or Humber College. Paul Cooling is President over about 60 members. For information regarding this club, please write to:

Ross Cooling
170 Redpath Avenue
Toronto, Ontario
Canada M4P 2K6

"Education regarding Microcomputers. Yearly dues and a monthly newsletter."

μ

Software for the Apple II

SUPER CHECKBOOK—a program designed to be an electronic supplement to your checkbook register. It's disk oriented and allows information to be displayed on the video screen or printer. It's super fast in sorting and retrieving information and totals. As an added bonus the program can optionally provide bar graphs to screen and/or printer. The program performs all standard check register operations, i.e. reconciliation. Minimum requirements are Disk II and only 32K RAM memory if Applesoft is in ROM; $19.95.

ADDRESS FILE GENERATOR—a program that gives you complete control over a name and address file at a very low price. The power and flexibility of this software system is unmatched even in programs costing much more. You are allowed up to eleven fields in each record and you can search and sort on any of these fields. In fact you can sort up to three fields at once. The program contains a powerful print format routine which allows you to print out any field in any format you wish. Minimum requirements are Disk II and only 32K RAM memory if Applesoft is in ROM; $19.95

WORLD OF ODYSSEY—an adventure game to which all others must be compared. It's by far the most complex game for the Apple II. It will probably drive you crazy and take several months of play to completely traverse this world. You have 353 rooms on 6 different levels to explore with myriads of treasures and dangers. The program allows you to stop play and to optionally save where you are so that you can resume play at a later time without having to repeat previous explorations. It's been called the best adventure game yet! Minimum requirements are Disk II with 48K RAM and Applesoft II in ROM; $19.95.

REAL ESTATE ANALYSIS PROGRAM—The Real Estate Analysis Program provides the user with three features. a) A powerful real estate investment analysis for buy/sell decisions and time to hold decisions for optimal rental/commercial investments. b) Generation of complete amorization schedules. c) Generation of depreciation schedules. All three features are designed for video screen or printer output. In addition, the program will plot; cash flow before taxes vs. years, cash flow after taxes vs. years, adjusted basis vs. years, capital gains vs. years, pre-tax proceeds vs. years, post-tax proceeds vs. years, and return on investment (%) vs. years. Minimum requirement Applesoft II, 16K; $14.95.

DYNAMAZE—a dazzling new real-time game. You move in a rectangular game grid, drawing or erasing walls to reflect balls into your goal (or to deflect them from your opponent's goal). Every ball in your goal is worth 100 points, but you lose a point for each unit of elapsed time and another point for each time unit you are moving. Control the speed with a game paddle: play as fast as ice hockey or as slowly and carefully as chess. Back up and replay any time you want to; it's a reversible game. Integer Basic (plus machine language); 32K; $9.95

ULTRA BLOCKADE—the standard against which other versions have to be compared. Enjoy Blockade's superb combination of fast action (don't be the one who crashes) and strategy (the key is accessible open space—maximize yours while minimizing your opponent's). Play against another person or the computer. New high resolution graphics lets you see how you filled in an area—or use reversibility to review a game in slow motion (or at top speed, if that's your style). This is a game that you won't soon get bored with! Interger Basic (plus machine language); 32K; $9.95.

What is a REVERSIBLE GAME? You can stop the play at any point, back up and then do an "instant replay", analyzing your strategy. Or back up and resume the game at an earlier point, trying out a different strategy. Reversibility makes learning a challenging new game more fun. And helps you become a skilled player sooner.

Available at your local computer store

Call or write for our free SOFTWARE CATALOG

Apple II is a registered trademark of Apple Computer, Inc.

DEALER INQUIRIES INVITED

POWERSOFT, INC.

P. O. BOX 157
PITMAN, NEW JERSEY 08071
(609) 589-5500

Programs Available on Diskette at $5.00 Additional

- Check or Money Order
- Include $1.00 for shipping and handling
- C.O.D. ($1.15 add'tl. charge)
- Master Charge and VISA orders accepted
- New Jersey residents add 5% sales tax

The MICRO Software Catalog: XXI

Software announcements for the 6502 based systems

Mike Rowe
P.O. Box 6502
Chelmsford, MA 01824

Name:	**ISAM-DS**
System:	**Apple II**
Memory:	**3K plus index table storage**
Language:	**Applesoft**
Hardware:	**Apple II, Disk II**

Description: ISAM-DS: integrated set of 15 utility routines: facilitate creation & manipulation of indexed files. Records on indexed files quickly retrieved randomly or in sequence. Each record identified by key data value which does not have to be part of record, doesn't have to be unique for each record. Partial key values may be used in retrieving records. Interface between ISAM-DS and an Applesoft program through single entry point (GOSUB) and 9 variables. Files created, opened, closed, copied, erased. Records written, read, changed, deleted. File space freed by deleting record automatically reused when another record added. No need to clean up file due to update activity.ISAM-DS must for writing business systems for the Apple II and equally useful in personal programs or learning index-sequential file processing techniques.

Copies:	**Just released**
Price:	**$50.00** (Texas residents add 5% sales tax.)
Includes:	integrated set routines, documentation for routines, & sophisticated mailing list program: demonstrates ISAM-DS capabilities. Append routines for DOS 3.1 and 3.2 also included. Append routines used to join the ISAM-DS package to an Applesoft program.
Author:	**Robert F. Zant**
Available:	Decision Systems P.O. Box 13006 Denton, TX 76203

Name:	**Zero Based Budgeting**
System:	**Apple II or Apple II plus. Printer optional.**
Memory:	**32K**
Language:	**ROM or cassette Applesoft**

Description: Program allows user to create zero based budget & store data on tape for later recall. 10 projects, 16 costs centers allowed. Independent $ rates for each cost cntrentered with man-yr entries for each specific combination. Once data entered, numerous modify routines are available in menu format for alteration of data. Project Priority feature included with accumulative expense listings show how much expense occurring as each project is added. Cost & human resource totals shown for each cost center & project independently. Particularly useful program for both sm & lg businesses. Contains useful features for personal budgeting. Hard copy output provided in software if suitable interface available.

Copies:	**Available now**
Price:	**$12.95 Postpaid**
Includes:	Cassette tape, loading instructions, description and example.
Author:	**Neil A. Robin**
Available:	Tech-Digit Co. 21 Canter Lane Sherwood, OR 97140

Name:	**Major League Baseball**
System:	**Apple II**
Memory:	**48K and ROM Applesoft**
Language:	**Applesoft**
Hardware:	**Disk**

Description: Manage Major League Baseball teams & make all decisions. Includes 1979 teams and utility programs to create and maintain team files. HIRES display.

Copies:	**Just released**
Price:	**$25.00**
Author:	**Stan Erwin**
Available:	5410 W. 20th Street Indianapolis, IN 46224

Name:	**Subroutine Library**
System:	**UK 101, based on OSI Challenger**
Memory:	**8K**
Language:	**Microsoft Basic and Machine Code**
Hardware:	**UK 101/OSI Challenger**

Description: Library of useful subroutines, for screen formatting. Fast genrlized histograms, vert (1/8 pixel res) and horiz (½ pixel), screen clear/fill with character instantly, instant blocks/lines of any character; place text anywhere on screen, horiz or vert; keybd control w/o 'INPUT'; save variables for chaining to new program; hex/decimal conv with error flags; etc. The UK 101 identical to Challenger but screen format is 48X16. Elementary convers of screen routines will be needed. Machine code routines for fast writing reside is top 256 bytes of memory. Further routines under development, e.g. save/read text on cassette. Also original games available.

Copies:	**25 sold in one month**
Price:	**$12.00**
Author:	**J. M. Leach**
Available:	Dola Software 117,Blenheim Road Deal, Kent, England

Name:	**MAILBAG**
System:	**Apple II or Apple II Plus**
Memory:	**32K with ROM Applesoft or the Language Card, 48K with RAM (disk) Applesoft**

Language:	**Applesoft and machine language**
Hardware:	**Apple II, Disk II (a printer with Serial or Parallel interface desirable. Supports the Paymar Lower Case Adapter chip)**

Description: a menu-driven, disk based mailing list database sys. Fully supports UPPER & lower case letters from stndard APPLE keybd, displays true l.c. when running with Paymar LCA, or with l.c.in INVERSE video otherwise. Provides easy-to-use method of screen formatting for data entry,right & left-arrow key editing. Dual-disk operation fully supported as is ability to do DOS CATALOG w/o leaving sys. Besides normal name, addr, & phone # lines, 2nd/company name available as is optional coments/code line. User may search database on any of 6 keys & sort entire data set on any of 6 sort keys. Sorting via fast Shell-Metzner sort in Basic, or MB can interface easily to utlra-fast mach lang sort program Ampersort II.Special feature: ability to automatically merge mail list with form letters created by TXT/ED 2.0 Word Proces Sys. Most popular printers may be easily infterfaced, printing options incl full database printouts, standard mail labels, special 1-line name & phone # list, + form letters.

Copies:	**Just Released**
Price:	**$34.99 on disk**
Includes:	System disk & documentation manual with instructions for printer interfacing ans sort method modifications.
Author:	**Gerald H. Rivers**
Available:	G.H. Rivers P.O. Box 833 Madison Heights, MI 48071

Name:	**ASTAT 79**
System:	**APPLE II or APPLE II Plus & Disk II**
Memory:	**32K with ROM—48K without**
Language:	**APPLESOFT II**

Description: An integrated statistical processor; subset of popular main-frame packg P-STAT 78. Calculates descriptive statistics (min, max, sum, mean, standard deviation, & counts), freq distributions (count, %, & cumulative %), complete bivariate tabulations (count, row column total %, cell expected-valued, cell Chi-square), correlation matrices (Pearson correlations, pair-wise deletion of missing data, good n's), path analysis(multiple regression Beta weights & R-square). Also contains complete variable transf lang. Logic driven algebracic file modifications can be permanent or applicable to specific procedures. Missing values understood by all procedures. Supports complete dat-base mangement functions such as formatted listings of files, alphabetic fields, file merging (up-down or left-right) & more. All files standard DOS text files.

Price:	**$100.00 complete — $20.00 each procedure.**
Includes:	One diskette plus sample program & language reference manual. Machine readable manual available at extra cost. User's manual available soon.
Author:	**Gary M. Grandon, Ph.D.**
Available:	Rosen Grandon Associates 296 Peter Green Road Tolland, CT 06084 or The Roper Center, Inc. University of Connecticut Box U-164R Storrs, CT 06268

Name:	**List CTRL**
System:	**Basic Apple II or Apple Plus**
Memory:	**32K or 48K**
Language:	**Assembly (.8K)**
Hardware:	**Disk II, Applesoft ROM**

Description: This utility allows you to control the listout of a program to the screen. The Apple game paddles are utilized to control an adjustable pause between each line of text and/or character outputted to the screen. The speed of text output to the screen can be changed at any time with the paddle controls to accommodate a comfortable reading level. The game paddle button can be pressed at any time during a listout. This will stop the listout and allow you to study the page of text, take notes, etc. By pressing the spacebar key, the next page of text (20 lines) wll be output to the screen and will stop. Each succeeding press of the spacebar will display the next page of text. If the game paddle button is pressed again, the page listout is diabled and the listout will continue at the speed set by the game paddles. The ESC key may be used at any time to stop the present listout and allow a new listout of the program; any portion of it, or you can exit List Ctrl. Upon exit, List Ctrl may be retained in memory; called back with the Ampersand key for later use; or aborted. Regardless of the usage, DOS is never disabled.

Copies:	**Many**
Price:	**$14.95**
Includes:	User Documentation and Diskette
Author:	**Frank D. Chipchase**
Available:	Soft Ctrl Systems P.O. Box 599 West Milford, NJ 07480

Name:	CHAT (Challenger Terminal)
System:	**OSI Challenger-1P and Superboard II**
Memory:	**4K**
Language:	**Machine Code and Basic**
Hardware:	**Modem and RS232 modification**

Description: An intelligent terminal with the ability to directly transmit data from cassette and transfer received data to cassette via an internal buffer. This buffer automatically expands on systems with more than 4K to allow more data storage at a single time. Full/half duplex modes; selectable parity and stop bits. CHAT has a very unique feature—46 user definable 6-state keys capable of generating all ASCII characters. The user may be content with the standard ASCII keyboard layout that CHAT comes with or may tailor it to his/her needs. All key changes are stored when CHAT is saved on tape. Also, the keyboard has the auto-repeat feature and a break key.

Price:	**$24.95**
Includes:	Cassette and User Manual
Author:	**Charles A. Shartsis** 9308 Cherry Hill Rd., No. 812 College Park, MD 20740

Name:	**FLEXIPAY**
System:	**APPLE II**
Memory:	**48K with ROM or RAM Applesoft**
Language:	**Applesoft**
Hardware:	**Apple II Disk II, Serial Interface printer**

Description: FLEXIPAY is a versatile payroll system for the APPLE II. Payroll masterfiles for any number of companies may be created, edited and fully maintained. Processes weekly, monthly, salaried, hourly, commissions, etc. Automatic group insurance deductions plus two other auto deducts (credit union, etc.) of your choice. All taxes computed. Pay and nonpay adjustments (advances, etc.). Output includes masterfile, payroll summary with current, QTD and YTD data for each employee, company totals. Labor summary, tax summary, departmental cost summary, plus worksheets for next pay period. Fast, easy entry, self-verified. Prints paychecks and stubs for 40 (RAM Applesoft) or 60 (ROM Applesoft) employees. Prints W-2's and 941's for state use. Formatted checks and stubs available—have your bank no. printed or we will handle. Start your own service bureau!

Price:	**$75.00**
Includes	System disk, full instructions.
Author:	**S. Prater, CPA**
Available:	INDATA, Inc. 8222 Antoine, suite 103 Houston, TX 77088

Name:	**Stock Market Option Account**
System:	**Apple II or Apple II Plus**
Memory:	**32K with Applesoft ROM** **48K with Applesoft RAM**
Language:	**Applesoft II**
Hardware:	**Disk II, 132 column printer (optional)**

Description: The program stores and retrieves virtually every option traded on all option exchanges. A self-prompting program allowing the user to enter short/long contracts. Computes gross and net profits/losses, and maintains a running cash balance. Takes into account any amending of cash balances such as new deposits and/or withdrawals from the account. Instantaneous readouts (CRT or printer) of options on file, cash balances, P/L statement. Includes routine to proof-read contracts before filing.

Price:	**$19.95 plus $2.00 (P&H) first class**
Includes:	Diskette and full documentation
Available:	Mind Machine, Inc. 31 Woodhollow Lane Huntington, NY 11743

Name:	**Program Writer**
System:	**Apple**
Memory:	**32K minimum**
Language:	**Applesoft**
Hardware:	**1 Disk Drive**

Description: This program was written to speed up the process of writing advanced business programs. It works as a data management system, but also writes disk statements as permanent line number, if requested. Supports 20 fields per entry, searching or sorting by any field, generating reports, packing numbers to increase disk space, plus many more. Use for inventory, checks, phone numbers, etc. Simple to use with instructions.

Copies:	**Just released**
Price:	**$29.95**
Includes:	One diskette, instructions and examples.
Author;	**Wilford Niepraschk** 59 Thurston Avenue Virginia Beach, VA 23455

Name:	Data Factory 2.2
System:	**Apple 1 or 2 disk drives or Hard disk (Corvus.Lobo) optional printer**
Memory:	**48K RAM ROM card Language card**
Language:	**Applesoft**

Description: A data base file program of unique utility. It allows the user to create a file consisting of desired catagories (columns or fields) in which various sorting and printing proceedures can be accomplished. The program can be copied, lists for modification. It uses one or two disk drives and operates with or without a printer. Printer options are on a separate program and allows you to select printer slot, line length, indentation, lines per page, line feeds for single or roll-fanfold paper. Page numbers can be printed, if desired. Thses printer options can be easily changed on a permanent or temporary basis.

Price:	**$100.00 (Hard disk version slightly more).**
Includes:	Disk, program and 26 page manual.
Author:	**William Passauer**
Available:	Andent Inc. 1000 North Ave Waukegen, IL 60085

Name:	**L.I.S.A. (Lazer Systems' Interactive Symbolic Assembler)**
System:	**APPLE II**
Memory:	**V1.5C: 48K, V2.0:64K**
Language:	**6502 Machine Language**
Hardware:	**Apple II Disk II, Language Card (V2.0), optional: Dan Paymar Lower Case mod, 80 column printer, Mountain Hardware Romwriter, (V2.0), Double Vision 80X24 display board (V2.0)**

Description: An interactive 6502 assembler for the Apple II mirocomputer. Syntax checkline is perfromed at edit time resulting in immediate feedback for all syntax and addressing mode errors.. Designed specifically for the Apple II, LISA incorporates several special features such as the ability to store data in inverted or blinking mode, built in disk operations, etc. LISA is the fastest assembler on any personal computer. At 20,000 lines per minute, (assembly time) it is four to te times faster than comparable assemblers on the Apple II. A typical 1000 line program only requires 3 seconds to assemble. Compare this to the 30 sceonds to three minutes required by other Apple assemblers to assmble the same file.

Price:	**$34.95, (V1.5C); $49.95 (V2.0)**
Includes	Diskette with software; 100 page documentation reference manual.
Author:	**Randall Hyde, Lazer Systems**
Available:	Programma International 3400 Wilshire Blvd. Los Angeles, CA 90010

Name:	**Soft-Sonic**
System:	**Apple II or Apple II Plus (except for speech program)**
Memory:	**32K**
Language:	**SS 6502 Assembly Language; Home control, timing—Applesoft; Home control, speech—Integer**
Hardware:	**Home Control, speech-Heuristics speech lab, Ultra sonic transducer and cable (included with programs) Disk highly recommended.**

Description: Three programs to provide a reliable and inexpensive means to interface a BSR (Sears) Home controller. SS is a reliable machine language subroutine that produces all the codes, tones and delays required to communicate with the BSR. Home Control, Speech provides for verbal control of up to ten lights and appliances with vocabularies for two persons automatically saved and exchanged from disk. Home control, timing has an internal software clock and allows for several hundred NAMED sequences to be executed or repeated so that 16 remotes may be turned off, on, dimmed, etc. automatically. Each sequence can be for several seconds or several weeks, depending upon your requirements. No modification of your Apple is required as the transducer (included) plugs into the game I/O.

Price:	**$39.95 plus $2.00 (P&H) GA residents add 4% sales tax.**
Includes:	SS Home Cntrl, spch, HC, timing, SS Relocator, all on disk (tape by request) and the ultra sonic transducer complete with cable.
Author:	John Blakenship
Available:	B.A.C.E. P.O. Box 52785 Atlanta, GA 30355

While we have been lenient in the past regarding the length of the entries in the Software Catalog, we must now insist that future entries be kept as brief as possible. We think that twelve to fifteen lines in the "description" part of the entry should keep it about right. The other parts, as long as needed.

We now have so many entries backed up, that we feel this policy is only fair to give everyone 'equal time'. We will be fored to edit, or return any entries that we judge too long.

Mike Rowe

HDE inc.

BOX 120
ALLAMUCHY, NJ 07820
201-362-6574

HUDSON DIGITAL ELECTRONICS INC.

COMING SOON!

omni 65

For 6502 Systems Development
Engineering Support
Word Processing Applications

The latest in a continuing series of advanced hardware and computer program products for KIM, AIM, TIM, SYM.

JOHNSON COMPUTER
Box 523, Medina, Ohio 44256
(216) 725-4560

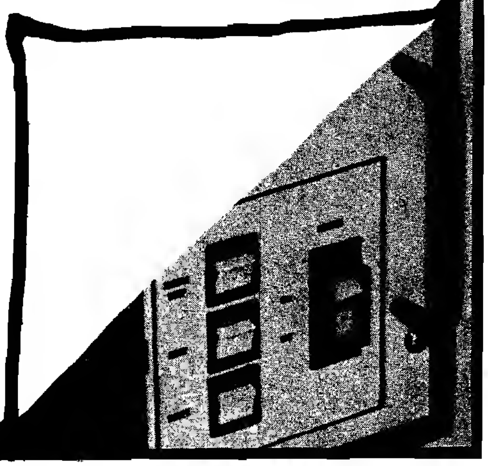
ARESCO
P.O. Box 43, Audubon, Pa. 19407
(215) 631-9052

PLAINSMAN MICROSYSTEMS
Box 1712, Auburn Ala. 36830
(800) 633-8724

FALK-BAKER ASSOCIATES
382 Franklin Ave., Nutley, NJ 07110
(201) 661-2430

PERRY PERIPHERALS
P.O. Box 924, Miller Place, NY 11764
(516) 744-6462

PROGRESSIVE COMPUTER
SYSTEMS
405 Corbin Rd., York, Pa. 17403
(717) 845-4954



DISCOUNT DATA PRODUCTS

BASF 5¼" DISKETTES: $34.50 PER BOX OF 10

HIGHEST QUALITY DISKETTES AT A BARGAIN PRICE! LABELS AND WRITE-PROTECT TABS INCLUDED.

VINYL DISKETTE HOLDERS FOR NOTEBOOKS

THE IDEAL WAY TO STORE DISKETTES. EACH VINYL PAGE HOLDS TWO DISKETTES AND INCLUDES A POCKET FOR EACH DISKETTE'S LABEL. SAFELY KEEP UP TO 40 DISKETTES IN A SINGLE 1" 3-RING NOTEBOOK!

$4.95/SET OF 10

MARKETING YOUR OWN SOFTWARE? DDP OFFERS DEALER & SOFTWARE HOUSE DISCOUNTS ON NOT ONLY THE ABOVE ITEMS, BUT ALSO THE FOLLOWING PRODUCTS:

9" x 12" ZIP-LOCK BAGS FOR PACKAGING & DISPLAY OF SOFTWARE.

CORRAGATED MAILERS TO SHIP TO USERS OR DEALERS!

SEND FOR FREE INFORMATION AT:

DISCOUNT DATA PRODUCTS
P.O. BOX 19674-M
SAN DIEGO, 92119

(ADD $1.00 SHIPPING/HANDLING CHARGE TO ALL ORDERS.)

KIMSI FLOPPY DISKS—

PERRY PERIPHERALS HAS THE HDE MINIFLOPPY TO KIMSI ADAPTER

MINIFLOPPY S-100 ADAPTER: $15 + 1.50 p&h
($20. as of June 1, 1980

- **FODS and TED Diskette**
- **FODS and TED User Manuals**
- **Complete Construction Information**

(Not a kit; no parts supplied)

OPTIONS:

- **FODS Bootstrap in EPROM (1st Qtr'80)**
- **HDE Assembler (ASM) $75**
- **HDE Text Output Processor (TOPS) $135**

(N.Y. State residents, add 7% Sales Tax)

Place your order with:

PERRY PERIPHERALS
P.O. Box 924
Miller Place, N.Y. 11764
(516) 744-6462

Your "Long Island' HDE Distributor

6502 Bibliography: Part XXI

Continuing bibliography of 6502 related material

Dr.William R. Dial
438 Roslyn Avenue
Akron, OH 44320

627. Stems from Apple 2, Iss. 12 (Dec. 1979) cont'd.

Hoggatt, Ken, "Ken's Korner," pg. 6-7.
Misc. notes on Pascal for the Apple.

Pell, George, "Comma's and Colons in Applesoft," pg. 7-8.
How to deactivate the "Extra Ignored" error message.

John, Norma M., "Pascal MASTERMIND," pg. 8-15.
Notes on Pascal and a program listing, MASTERMIND.

628. MICRO, No. 19 (Dec. 1979)

Brady, Virginia Lee, "Data Statement Generator," pg. 5-7.
An Apple program which writes its own DATA statements.

Figueras, John, "How to do a Shape Table Easily and CORRECTLY," pg. 11-22.
A tutorial and program to create shape tables, for the Apple.

Tulloch, Michael, "Relocating PET BASIC Programs," pg. 25-27.
Discussion of the organization of PET BASIC and a relocation program.

Swindell, Jack Robert, "If You Treat It Nicely, It Won't Byte," pg. 31-34.
Discussion of the OSI Superboard II.

Babcock, Robert E., "Sharpen Your AIM," pg. 37-39.
Four programs to enhance the capabilities of the basic AIM 65.

Erler, Kevin, "An Additional I/O Interface for the PET," pg. 40-41.
Interfacing a VIA 6522 to your PET is simple.

Kolbe, Werner, "A 60 X 80 Life for the PET." pg. 45-47.
A bigger display for your PET when playing the Game of Life.

Guild, George S., Jr., "Applesoft Program Relocation," pg. 49.
A simple technique to change the program storage space when using Applesoft.

Welch, Steven M., "KIM and SYM Format Cassette Tapes on Apple II," pg. 51-56.

Taylor, William L., "Graphics and the Challenger 1P," pg. 61-65.
Discussion of graphics on the OSI C1P microcomputer.

Suchyta, Casmir J., III and Zitzewitz, Paul W., "Time of Day Clock and Calendar for the SYM-1," pg. 67-68.
Have a clock and calendar running in your SYM at the same time you are running a program.

Broderick, John, "Apple II Speed Typing Test with Input Time Clock," pg. 69.
Care to take a speed typing test on your Apple???

Mitchell, S. Felton, Jr., "SUMTEST: A Memory Test Routine forthe6502," pg. 73-74.
No microcomputer is better than its RAM memory. Here is a test for any 6502 system.

Rowe, Mike, (Staff), "The MICRO Software Catalogue: XV," pg. 75-76.
Nine new programs are reviewed.

Dial, William R., "6502 Bibliography: Part XV," pg. 77-78.

629. 73 Magazine No. 231 (Dec. 1979)

Creason, Sam, "Teaching Your Micro to Count," pg. 104-113.
Two methods for adding counter capability to your 6502 machine. Hardware and software.

630.BYTE 4, No. 12 (Dec. 1979)

A. Osborne/McGraw Hill, 630 Bancroft Way, Berkeley, CA 94710
New book, Lance Leventhal, "6502 Assembly Language Programming," $12.50.

631. Abacus Newsletter 1, No. 12 (Dec. 1979)

Lowe, Stanley, "Inventory Formulas," pg. 3-5.
Inventory program for the Apple II.

Lowe, Stanley, "Marketing Formulas," pg. 6-8.
An Applesauce program for the Apple.

Lowe, Stanley, "Price Level Adjustments," pg. 9-10.
A business program for the Apple.

Anon, "Inserting Line #65535 in Integer Basic,;; pg. 9-10.
A useful Apple Utility routine.

Gauthier, Joe, "Report from Joe," pg. 11.
Program for generating "officialese."

632. Compute, Iss 2 (Jan./Feb. 1980)

Hulon, Rick and Belinda, "Sorting Sorts: Part 2," pg. 11-16.
A comparison of HEAP SORT and QUICK SORT.

Herman, Harvey B., "Memory Partition ofBASIC Workspace," pg. 18-20.

Baker, Robert W., "An Easier Method of Saving Data Plus Home Accounting," pg. 23-27.
Home Accounting on the PET.

Lindsay, Len, "Word Processors: A User Manual of Reviews," pg. 29-34.
This concludes part two of an overview of current word processing programs for the PET.

Butterfield, Jim, "Book Review:6502 Assembly Language Programming—by Lance A. Leventhal," pg. 36.

DeJong, Marvin L., "Machine Language Versus Basic Prime Number Generation," pg. 39-40.
A program to calculate prime numbers on the PET.

Pratto, Marlene, "The Learning Lab," pg. 41.
A list of educational PET programs by categories.

Earnhardt, Don, "A Printer for the Apple: The Heath H14," pg. 66.
An evaluation of the Heath H14 Printer interfaced to the Ap-

ple.

Klepfer, Joretta, "Atari BASIC and PET Microsoft BASIC," pg. 70.
A comparison of two versions of Basic and the tutorial manuals.

Victor, John,"Atari Basic," pg. 76-77.
Discussion of Graphics, Sound, Control Characters, Editing.

Bunker, W.M., "Lower Case Descention on the Commodore 2022 Printer," pg. 8l.
How to create more readable lower case letters on the 2022.

Richter, Mike, "Saving Money in Large Programs," pg. 82.
Hints for conserving memory on the PET.

Butterfield, Jim, "The Deadly Linefeed," pg. 82.
Precautions to use on the PET.

Stuart, Chuck, "Using Direct Access Files With the Commodore 2040 Dual Drive Disk," pg. 87-89.
Part two of a continuing series.

Matsumoto, Yashiko; Weinshank, Donald; Davis, Harvey, "Null Return ('LINPUT') Simulation for PET Users," pg. 90-91.
Touch return to go on in this subroutine.

Butterfield, Jim, "A Few Entry Points, Original/Upgrade ROM," pg. 93.
A listing of entry points.

Lindsay, Len, "Plotting With the CBM 2022 Printer," pg. 93-94.
How to plot graphics with the PET printer.

Butterfield, Jim, "Inside the 2040 Disk Drive," pg. 94-95.
Discussion of how the disk works.

Rehnke, Eric, "The Single-Board 6502," pg. 102-106.
Comparison of the KIM, AIM, SYM and SUPERKIM.

Mackay, A.M., "SYM-1 Message Scroller," pg. 108.

Stanford, Charles L., "Adapting BASIC Programs for Other Computers to the Challenger 1P," pg. 110-112.

Kelley, Ralph, "Proofread," pg. 112.
An efficient proofreading routine for the KIM.

DeJong, Marvin L., "Two Notes on the Pulse Counting Mode of Timer 2 on the 6522," pg. 114.

Beal, Barry L., "Tokens in OSI BASIC," pg. 116.
Misc. notes on OSI Tokens, etc.

633. Electronic Engineering Times, Jan. 7, 1980.

DeSantis, Tom, "Low-Cost IEEE-488 Systems Using the Commodore PET Microcomputer.," pg. 28-29.
The PET is being used extensively as a low-cost IEEE-488 bus controller.

634. Apple-Com-Post, No. 5 (Dec. 1979)

Anon, "Tips and Tricks," pg. 7-9.
Several short routines, including a test to indicate if a program is Present, Change of Register Content, DOS Identifier, Reading Random Text Files, Move, etc.

Anon, "Pascal," pg. 10.
Discussion of Pascal.

635. Cider Press 2, No. 6 (Dec. 1979).

Anon, "December DOM," pg. 4.
The December Disk of the Month with be a great Holiday DOM and will contain Bruce Tognazzni's CATTLE CAR.

Nareff, Max J., "Avoiding the 'GET' Trap," pg. 5.
Trace with the DOS up on the Apple and also use a DOS command directly following a GET statement.

Nareff, Max J., "Make a Box," pg. 5.
Boxes for program headings on Apple programs.

Vrooman, Gerry, "The Defogger Finds Illegal Characters," pg. 6.
How to trick your Apple into accepting illegal commands.

Hyde, Randy, "Lisa Author Strikes Back," pg. 7.
Further discussion of theLISA Assembler, for the Apple.

Anon, "Simple Tones for Applesoft II," pg. 8.
Simple routine modified from the "Red Book" for Applesoft.

Carlisle, Rod, "Put the HEX on Hexadecimal," pg. 9.
Tables for converting Hexadecimal to Decimal.

Wilson, Gene, "Pascal-Single Drive," pg. 12.
No doubt you need help If your are trying to run Pascal with only one disk drive. Well, here it is!

Fields, Randy, "Apple Owners' Questionaire Results," pg. 13.
Results of a very interesting survey of Apple owners.

Apple Computer Staff, "Application Note 1," pg. 14-15.
Description of text pages, text screen maps, character display values, etc.

636. Southeastern Software Newsletter, Iss 15 (Dec. 1979)

McClelland, George, "Review of Text Editors/Word Processors," pg. 1-11.
Includes Easy Writer, Text Editor (Peripherals), Super-Text, Big-Edit, Apple P.I.E. 2.0 and Format and comparisons to Dr Memory, and earlier editors.

637. The Seed 1, No. 7 (Dec. 1979)

Foens, Bob, "GEEJO," pg. 4.
Two Holiday season programs for the Apple.

Thompson, Colleen, "Hunt the Wumpus—Revisited," pg. 6-8.
Hunt the Wumpus with crooked arrows which riccochet—a new version of an old game.

Wagner, Roger, "A Fast GR Screen Clear," pg. 9.
A program utilizing the Monitors binary move routine which clears the low resolution screen very quickly, on the Apple.

Knaster, Scott, "Solving the RESET Problem," pg. 14.
Noone likes the RESET Key! New uses make it lovable!

Knaster, Scott, "The Amplified Apple," pg. 16-17.
A discussion of the "mystery key," the Ampersand. This key on the Apple can be very useful. For instance, the new command—READAT.

Wagner, Roger, "An Unlikely Character," pg. 18.
How to generate some real unlikely special characters on the Apple keyboard using fancy fingering.

638. Creative Computing 5, No. 12 (Dec. 1979)

Heuer, Randy, "Satellite Tracking Software," pg. 32.
Review of a new package by SAT TRAK INYL.

Waite, Mitchell, "Animation for the Apple," pg. 126-128.
Walking animation made simple on the Apple.

Carpenter, Chuck, "Apple-Cart," pg. 141-144.
A Pascal TURTLEGRAPHICS PROGRAM, Assembly Language Program for a clock routine, etc.

Yob, Gregory, "Personal Electronic Transactions," pg. 146-149.
Review of Hal Chamberlin's D/A Board from Micro Technology, Unltd. Real music from the PET at last. Also this month: What to do about "Out-of-Memory" errors.

639. Appleseed Newsletter (San Antonio) 2, No. 1 (Jan. 1980)

Wright, Don, "Underground Movement," pg. 1.
A discussion of Conflict Simulations (Gaming(with references to RISK, Global War, Warlords, Wilderness Campaign, Dungeon Monster, Metamorphosas Alpha, Boot Hilland others.

The Paper 2,Iss 10 (Jan. 1980)

Swan, Warren D., "PET 2022 Tractor Feed Printer," pg. 9-15.
A thorough product review and also includes some software routines for the PET printer.

Anon, "Machine Language Programming," pg. 16-19.
Discussion of the PET USR function, PET number representation, fixed point numbers, etc.

Busdiecker, Roy, "Power-Root," pg. 19-20.
A program involving roots.

McArthur, James F., "SEARCH," pg. 24-25.
Search through your PET program for a given line using this routine.

1Stetzer, Stephen, "Combined Budget," pg. 25-27.
Keep current on certain expense items of recurring interest.

MICRO, No. 20 (Jan. 80)

Lacy, Allen J., "Tape Execute File Create and Use," pg. 5-7.
How to convert Integer Basic Apple programs to Applesoft Basic without a disk.

Beach, Bruce M., "Why a PET, Apple, 6502 BASIC Compiler? A Simple Explanation," pg. 9-12.
What a Compiler is, how it works and a discussion of a Basic compiler currently under development.

Reich, Dr. L. S., "Human Pysiological Parameters," pg. 15-19.
Program calculates the proper weight for an individual as a function of height, body build and sex. In Applesoft.

DeJong, Marvin L., "Lifetime of a Non-Renewable Resource," pg. 21-22.
A good model of an interactive Basic simulation.

Vile, Richard C., Jr., "Sweet-16 Programming Using Macros," pg. 25-29.
Info about MACROs in general, the Apple II Sweet-16 Interpreter and how to use them together.

Baxter, B.E., "Screen Write/File Routine," pg. 30-31.
A routine which makes it simple to Edit the Apple screen and save the screen image on disk.

Gieryic, Jack, "SYM-1 Tape Verification," pg. 35-37.
Check to see if your data has been recorded properly in your audio cassette.

Evans, Mel, "Symbol Table Sorter/Printer for the AIM Assembler," pt. 43-48.

Staff, "The MICRO Software Catalogue:XVI," pg. 51.
Five new programs are reviewed.

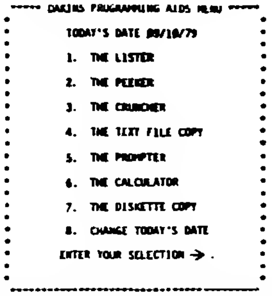
Childress, J.D., "Search/Change in Applesoft," pg. 55-58.
Search a file for a particular string and then change the string to a new one.

Peck, Robert A., "SYM-1 Staged Loading Technique for Segmented Programs," pg. 59.
How to load a continuous block 0000 to 03FF.

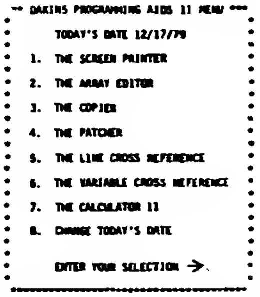
Dial, William R., "6502 Bibliography: Part XVI," pg. 61-62.
About eighty new references are covered.

μ

DAKIN5 UTILITIES



```
----- DAKIN5 PROGRAMMING AIDS MENU -----

      TODAY'S DATE 08/10/79

      1.  THE LISTER
      2.  THE PEEKER
      3.  THE CRUNCHER
      4.  THE TEXT FILE COPY
      5.  THE PROMPTER
      6.  THE CALCULATOR
      7.  THE DISKETTE COPY
      8.  CHANGE TODAY'S DATE

     ENTER YOUR SELECTION -> .
```



```
-- DAKIN5 PROGRAMMING AIDS II MENU ---

      TODAY'S DATE 12/17/79

  1.  THE SCREEN PRINTER
  2.  THE ARRAY EDITOR
  3.  THE COPIER
  4.  THE PATCHER
  5.  THE LINE CROSS REFERENCE
  6.  THE VARIABLE CROSS REFERENCE
  7.  THE CALCULATOR II
  8.  CHANGE TODAY'S DATE

      ENTER YOUR SELECTION ->.
```

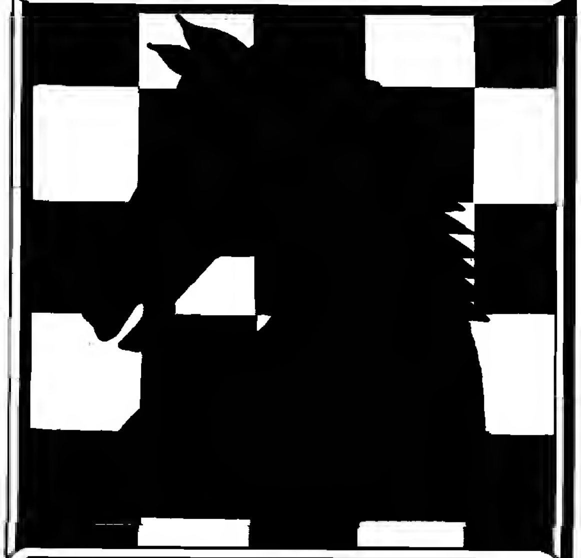
The Cruncher removes REM statements and compresses code in Applesoft programs. The Prompter is a powerful data entry subroutine that can handle both string and numeric data. Options include using commas. decimal points, and leading zeros with right-justified numerics. A maximum field length can be specified to prevent overflow in both numeric and alphanumeric fields. The Diskette Copy formats an output disk, copies each track, and verifies that the output matches the input. PLUS FOUR MORE UTILITIES TO AID YOUR OWN PROGRAMMING. Suggested Retail Price for Dakin5 Programming Aids is $39.95.

The Copier copies absolutely any kind of file or program from one diskette to another. The Variable Cross Reference produces a cross-reference for all variable names used in an Applesoft BASIC program. The Line Cross Reference creates a cross-reference for an Applesoft BASIC program, showing where a given line is referenced by GOTO, GOSUB, THEN, or LIST statements. The Patcher allows you to display any sector of a diskette, and then to update any data within that sector. PLUS THREE MORE UTILITIES TO FACILITATE YOUR OWN PROGRAMMING. Suggested Retail Price for Dakin5 Programming Aids II is $49.95.

Each utility package includes a program diskette and very comolete documentation. The hardware requirements are an Apple II, 48K of memory, 2 Disk II's, and a printer. Languages are Applesoft/Assembler.

See your Apple dealer or contact Dakin5 Corporation, P.O. Box 21187, Denver, Colorado 80221. Telephone:(303) 426-6090. Dakin5 developed the business application software The Controller™ for Apple Computer Inc.

MICROCHESS
for the AIM



MICRO SOFTWARE

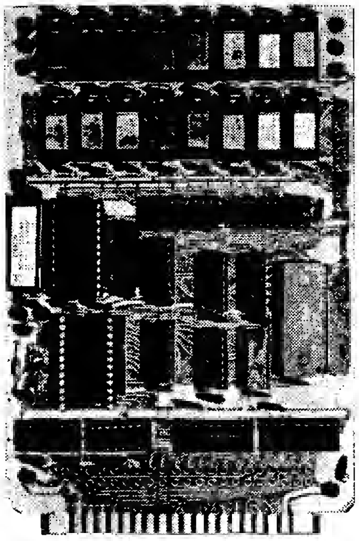
AIM MicroChess with Player's and Programmer's Manual, complete Source Listings, Object on Cassette Tape. $15.00 plus shipping [$1.00 US/$2.00 Anywhere Else]

MICRO Software, P.O. Box 6502, Chelmsford, MA 01824

KIM/SYM/AIM-65—32K EXPANDABLE RAM

DYNAMIC RAM WITH ON BOARD TRANSPARANT REFRESH THAT IS COMPATIBLE WITH KIM/ SYM / AIM-65 AND OTHER 6502 BASED MICROCOMPUTERS.

- ★ PLUG COMPATIBLE WITH KIM/SYM/AIM-65. MAY BE CONNECTED TO PET USING ADAPTOR CABLE. SS44-E BUS EDGE CONNECTOR.
- ★ USES +5V ONLY (SUPPLIED FROM HOST COMPUTER BUS). 4 WATTS MAXIMUM.
- ★ BOARD ADDRESSABLE IN 4K BYTE BLOCKS WHICH CAN BE INDEPENDENTLY PLACED ON 4K BYTE BOUNDARIES ANYWHERE IN A 64K BYTE ADDRESS SPACE.
- ★ ASSEMBLED AND TESTED BOARDS ARE GUARANTEED FOR ONE YEAR, AND PURCHASE PRICE IS FULLY REFUNDABLE IF BOARD IS RETURNED UNDAMAGED WITHIN 14 DAYS.
- ★ BUS BUFFERED WITH 1 LS TTL LOAD.
- ★ 200NSEC 4116 RAMS.
- ★ FULL DOCUMENTATION



ASSEMBLED/ TESTED	WITH 32K RAM	$419.00
	WITH 16K RAM	$349.00
	WITHOUT RAM CHIPS	$279.00
HARD TO GET PARTS ONLY (NO RAM CHIPS)		$109.00
BARE BOARD AND MANUAL		$49.00

PET INTERFACE KIT $49.00

CONNECTS THE ABOVE 32K EXPANDABLE RAM TO A 4K OR 8K PET. CONTAINS EXPANSION INTERFACE CABLE, BOARD STANDOFFS. POWER SUPPLY MODIFICATION KIT AND COMPLETE INSTRUCTIONS.

6502. 64K BYTE RAM AND CONTROLLER SET

MAKE 64K BYTE MEMORY FOR YOUR 6800 OR 6502. THIS CHIP SET INCLUDES:

- ★ 32 MSK 4116-3 16KX1, 200 NSEC RAMS
- ★ 1 MC3480 MEMORY CONTROLLER.
- ★ 1 MC3242A MEMORY ADDRESS MULTIPLEXER AND COUNTER.
- ★ DATA AND APPLICATION SHEETS. PARTS TESTED AND GUARANTEED.

$295.00 PER SET

16K X 1 DYNAMIC RAM

THE MK4116-3 IS A 16,384 BIT HIGH SPEED NMOS, DYNAMIC RAM THEY ARE EQUIVALENT TO THE MOSTEK, TEXAS INSTRUMENTS, OR MOTOROLA 4116-3.

- ★ 200 NSEC ACCESS TIME, 375 NSEC CYCLE TIME
- ★ 16 PIN TTL COMPATIBLE.
- ★ BURNED IN AND FULLY TESTED
- ★ PARTS REPLACEMENT GUARANTEED FOR ONE YEAR

$8.50 EACH IN QUANTITIES OF 8

BETA
COMPUTER DEVICES

1230 W.COLLINS AVE.
ORANGE, CA 92668
(714) 633-7280

[illegible]

ALL ASSEMBLED BOARDS AND MEMORY CHIPS CARRY A FULL ONE YEAR REPLACEMENT WARANTY

Missing MICRO Information?

MICRO is devoted exclusively to the 6502. In addition, it is aimed at useful, reference type material, not just "fun and games". Each month MICRO publishes application notes, hardware and software tutorials, a continuing bibliography, software catalog, and so forth. Since MICRO contains lots of reference material and many useful program, most readers want to get the entire collection of MICRO. Since MICRO grew very rapidly, it quickly became impractical to reprint back issues for new subscribers. In order to make the older material available, collections of the reprints have been published.

[A limited number of back issues are still available from number 7 to 18 and 20 to current. There are no 19's left.]

The BEST of MICRO Volume 1 contains all of the significant material from the first six issues of MICRO, from October/November 1977 through August/September 1978. This book form is 176 pages long, plus five removeable reference cards. The material is organized by microcomputer and almost every article is included. Only the ads and a few 'dated' articles have been omitted. [Now in third printing!]
Surface...$7.00 **Air Mail...$10.00**

The BEST of MICRO Volume 2 covers the second six issues, from October/November 1978 through May 1979. Organized by microcomputer, this volume is 224 pages long.
Surface...$9.00 **Air Mail...$13.00**

The BEST of MICRO Volume 3, covering the twelve issues from June 1979 through May 1980, will be over 400 pages long. It is scheduled for late summer 1980. The price is still to be determined.

For a free copy of the Index for Volumes 1, 2, and 3, please send a self-addressed, stamped envelope to:
BEST of MICRO, P.O. Box 6502, Chelmsford, MA 01824

Advertisers' Index

Aardvark	46
AB Computers	51
Beta Computer Devices	78
CJM-Industries	18
Classified Ads	55,56
The Computerist, Inc.	27
Computer Shopper	4
Dakin 5	78
DCS Software	46
Decision Systems	9
Discount Data Products	74
Electronic Specialists, Inc.	58
Enclosures Group	28
Galaxy	9
Holtzman	17
Hudson Digital Electronics	74
I & I Inc.	67
Instant Software	40,41
MICRO	78
Micro Austin	58
Micro Software	78
Muse	80
NIBBLE	14
On Line	36
Orion Software Associates	46
OSI Small Systems Journal	42-45
OSI	BC
Perry Peripherals	74
Powersoft, Inc.	70
Programma International	IBC
Progressive Computing	46
Progressive Software	62,63
Scelbi Publications	6
Shepardson Microsystems	67
SKYLES Electric Works	IFC,10,52
Small Business Computer Sys.	61
Southeastern Software	22
Strategic Simulations	39
Sybex	1
Systems Design	34
Systems Formulate	2
TEK Aids	51

SUPER-TEXT™

The Professional Word Processor

for the Apple II
and the Apple II plus

MUSE™
THE LEADER IN QUALITY SOFTWARE

STANDARD FEATURES

- single key cursor control
- automatic word overflow
- character, word and line insertion
- forward and backward scrolling
- automatic on screen tabbing
- single key for entering "the"
- auto paragraph indentation
- character, word and line deletion
- ditto key
- multiple text windows
- block copy, save and delete
- advanced file handling
- global (multi-file) search and replace
- on screen math and column totals
- column decimal alignment
- chapter relative page numbering
- complete printer tab control
- line centering
- superscripting and subscripting
- displays UPPER and lower case on the screen with Dan Paymar's Lower Case Adapter

FAST EDITING

Super-Text was designed by a professional writer for simple, efficient operation. A full floating cursor and multiple text screens facilitate editing one section of text while referencing another. Super-Text's advanced features actually make it easier to operate, allowing you to concentrate on writing rather than remembering complicated key sequences.

FLOATING POINT CALCULATOR

A built in 15 digit calculator performs on-screen calculations, column totals and verifies numeric data in statistical documents.

EXCLUSIVE AUTOLINK

Easily link an unlimited number of on-line files on one disk or from disk to disk. Autolink allows you to search or print all on-line files with a single command. Typical files of items that can be stored in this way include personnel files, prospect files, maintenance records, training records and medical histories.

ADVANCED FILE HANDLING

Single key file manipulation and complete block operations allow the user to quickly piece together stored paragraphs and phrases. Text files are listed in a directory with a corresponding index for fast and accurate text retrieval.

PRINTER CONTROLS

Super-Text is compatible with any printer that interfaces with an Apple. Print single or multiple copies of your text files or link files and they will be automatically printed in the specified order. User defined control characters can activate most special printer functions.

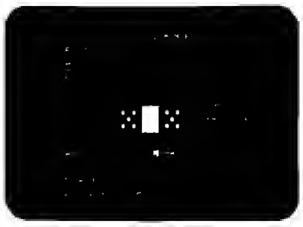
MODULAR DESIGN

This is a modularly designed system with the flexibility for meeting your future word processing needs. The first add-on module will be a form letter generator for matching mailing lists with Super-Text form letters.

SUPER-TEXT, requires 48K ($99.95)
Available TODAY at Computer Stores
nationwide. Dealer inquiries welcome. For more
information write:

MUSE SOFTWARE 330 N. Charles Street Baltimore, MD 21201 (301) 659-7212

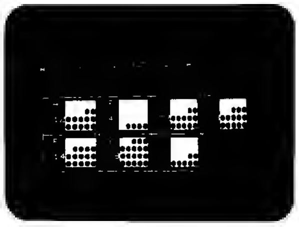
Software for the PET

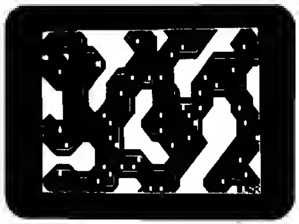

DOMINOES $ 6.95

OTHELLO $ 9.95

LETTER SQUARES $ 6.95

RPN MATHPACK $19.95

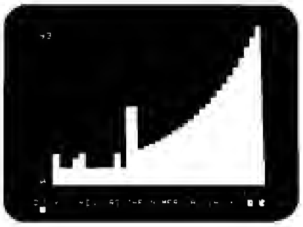

SUPER NIM $ 6.95


DIR/REF $ 6.95

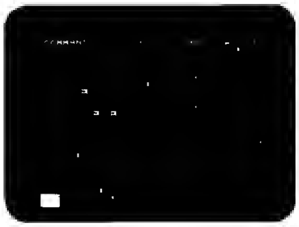
SPACE WARS $ 9.95

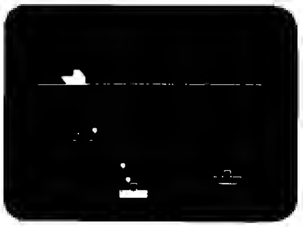
GRAND PRIX $ 6.95

E.S.P. $ 9.95


FORECAST $ 9.95

FOOTBALL $ 6.95


BLOCKADE $ 9.95


DEPTH CHARGE $ 9.95

BASKETBALL $ 9.95

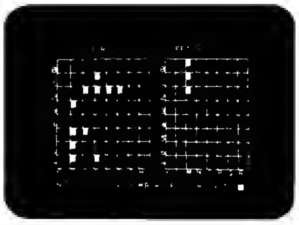
SLOT MACHINE $ 6.95

HOME ACCOUNTING $ 9.95

All orders include 3% postage and handling with a minimum of $1.00. California residents include 6% Sales Tax.

VISA MASTERCHARGE

PET IS A TRADEMARK OF
COMMODORE BUSINESS MACHINES, INC.


BATTLE SHIP $ 9.95

SHOOTING GALLERY $ 9.95

PROGRAMMA
INTERNATIONAL, Inc.
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579
384-1116
384-1117

Dealer Inquiries Invited

ZAP $ 6.95

PROGRAMMA Software Program Products

The home computer you thought was years away is here.

C8P DF $2,895

Ohio Scientific's top of the line personal computer, the C8P DF. This system incorporates the most advanced technology now available in standard configurations and add-on options. The C8P DF has full capabilities as a personal computer, a small business computer, a home monitoring security system and an advanced process controller.

Personal Computer Features

The C8P DF features ultra-fast program execution. The standard model is twice as fast as other personal computers such as the Apple II and PET. The computer system is available with a GT option which nearly doubles the speed again, making it comparable to high end mini-computer systems. High speed execution makes elaborate video animation possible as well as other I/O functions which until now, have not been possible. The C8P DF features Ohio Scientific's 32 x 64 character display with graphics and gaming elements for an effective resolution of 256 x 512 points and up to 16 colors. Other features for personal use include a programmable tone generator from 200 to 20KHz and an 8 bit companding digital to analog converter for music and voice output, 2-8 axis joystick interfaces, and 2-10 key pad interfaces. Hundreds of personal applications, games and educational software packages are currently available for use with the C8P DF.

Business Applications

The C8P DF utilizes full size 8″ floppy disks and is compatible with Ohio Scientific's advanced small business operating system, OS-65U and two types of information management systems, OS-MDMS and OS-DMS. The computer system comes standard with a high-speed printer interface and a modem interface. It features a full 53-key ASCII keyboard as well as 2048 character display with upper and lower case for business and word processing applications.

Home Control

The C8P DF has the most advanced home monitoring and control capabilities ever offered in a computer system. It incorporates a real time clock and a unique FOREGROUND/BACKGROUND operating system which allows the computer to function with normal BASIC programs at the same time it is monitoring external devices. The C8P DF comes standard with an AC remote control interface which allows it to control a wide range of AC appliances and lights remotely without wiring and an interface for home security systems which monitors fire, intrusion, car theft, water levels and freezer temperature, all without messy wiring. In addition, the C8P DF can accept Ohio Scientific's Votrax voice I/O board and/or Ohio Scientific's new universal telephone interface (UTI). The telephone interface connects the computer to any touch-tone or rotary dial telephone line. The computer system is able to answer calls, initiate calls and communicate via touch-tone signals, voice output or 300 baud modem signals. It can accept and decode touch-tone signals, 300 baud modem signals and record incoming voice messages. These features collectively give the C8P DF capabilities to monitor and control home functions with almost human-like capabilities.

Process Controller

The C8P DF incorporates a real time clock, FOREGROUND/BACKGROUND operation and 16 parallel I/O lines. Additionally a universal accessory BUS connector is accessible at the back of the computer to plug in additional 48 lines of parallel I/O and/or a complete analog signal I/O board with A/D and D/A and multiplexers.

Clearly, the C8P DF beats all existing small computers in conventional specifications plus it has capabilities far beyond any other computer system on the market today.

C8P DF is an 8-slot mainframe class computer with 32K static RAM, dual 8″ floppies, and several open slots for expansion.

C8P $950

Or get started with a C8P with cassette interface, 8K BASIC-in-ROM which includes most of the features of the C8P DF except the real time clock, 16 parallel I/O lines, home security interface and accessory BUS. It comes with 8K static RAM and Ohio Scientific's ultra-fast 8K BASIC-in-ROM. It can be expanded to a C8P DF later. Base price $950. Virtually all the programs available on disk are also available for the C8P cassette system on audio cassette.

Computers come with keyboards and floppies where specified. Other equipment shown is optional.

For literature and the name of your local dealer, CALL 1-800-321-6850 TOLL FREE.

OHIO SCIENTIFIC
1333 SOUTH CHILLICOTHE ROAD
AURORA, OH 44202 • (216) 831-5600